OPTIMIZATION MODELS AND ANALYSIS OF TRUCK-DRONE HYBRID ROUTING FOR LAST MILE DELIVERY

by

Patchara Kitjacharoenchai

A Dissertation

Submitted to the Faculty of Purdue University

In Partial Fulfillment of the Requirements for the degree of

Doctor of Philosophy

School of Industrial Engineering

West Lafayette, Indiana

May 2020

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF COMMITTEE APPROVAL

Dr. Seokcheon Lee, Chair

School of Industrial Engineering

Dr. Mario Ventresca

School of Industrial Engineering

Dr. Patrick Brunese

School of Industrial Engineering

Dr. Byung-Cheol Min

School of Computer and Information Technology

Approved by

Dr. Abhijit Deshmukh

Head of the Graduate Program

ACKNOWLEDGMENTS

This dissertation could not be accomplished without the help, support, and guidance of many people during my Ph.D. study.

First, I would like to express my deepest appreciation and gratitude to Dr. Seokcheon Lee, my major advisor for his invaluable guidance and full support over the past five years. I am grateful for his time and effort to train me to do good research which I can never achieve by myself alone. I am very fortunate to work with Dr. Lee under his guidance on things I am passionate about and he always gives me the flexibility to conduct research in my style. It is my honor to be his Ph.D. student in the Distributed Control Lab (DCLab) at Purdue University.

I would also like to thank my dissertation committee members who helped me improve the quality of the dissertation as follows: Dr. Mario Ventresca for his invaluable advice in helping me solve technical research problems as well as provide many insightful ideas to my study and career goal; Dr. Patrick Brunese for his comprehensive advice and encouragement to put all ideas into a research paper which enhanced my publication experience during my study; Dr. Byung-Cheol Min for inspiring and contributing many interesting ideas which turned into great research questions.

I am grateful to Mr. Dave Kotterman for providing funding resource through Purdue University Technical Assistance Program (TAP) for my Ph.D. study, Ms. Anita Park for helping me handle administrative tasks and the staff members in the School of Industrial Engineering at Purdue University for providing all facilities on campus which enhanced my learning experience and helped me reach the fullest potential in every aspect.

Special thanks to Dr. Mohammad Moshref-Javadi (MJ), Dr. Chul Hun Choi (Chris), Dr. Ashutosh Nayak, Dr. Siyu, Dr. Yang Zhang, Mr. Ho Young Jeoung, Mr. Rahul Sucharitha, Mr. Sungbum Jun, Mr. Zekun (Louis), Ms. Cansu Agrali, and the rest of the DCLab members for offering their

constructive feedback, sharing ideas, and providing their valuable friendship throughout my entire Ph.D. journey. I really did have fun and a great time with you all.

I am deeply indebted to my parents, Pichet and Yiamsiri, and siblings, Peerasarn, Piriya and Patara, for their full support and encouragement to pursue my education in the United States of America. Although we lived very far in distance, we had always remained very close and always supported each other. I would not have made this happen without them. Thanks also to Uncle Ekachai, Aunt Siriporn, Aunt Kanchasri and the Panitchewakul family, Aunt Kanchana and Uncle Surapon Pakviwat, Uncle Derrick, Aunt Sirirad and the rest of the Streng family for their emotional support and a warm cozy place to stay during the school break. I am very lucky to have them by my side in every moment.

Special thanks to Dr. Nuttida Yongsanguanchai, Apichaya Sangiamputtakoon, and my brother, Patara Kitjacharoenchai, for proofreading this dissertation.

Last but not least, I would like to thank my Thai friends, Dr. Nuttida Dr. Nattika, Dr. Benchapol, Dr. Theerayu, Dr. Puwadol, Dr. Peerapol, Dr. Gamonmas, Dr. Anirut, Dr. Kittipong, Dr. Nattawat, Dr. Bowornpath, Mr. Atipat, Mr. Thodsapol, Mr. Theerapat, Mr. Chod, Mr. Purin, …, and Thai Purdue Grad for their encouragement and support along these years.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ADI	Adaptive Insertion Heuristics
CVRP	Capacitated Vehicle Routing Problem
DRP	Drone Routing Problem
DTRC	Drone Truck Route Construction
FSTSP	Flying Side Kick Traveling Salesman Problem
GA	Genetic Algorithm
GRASP	Greedy Randomized Adaptive Search Procedure
I-VRPD	Integrated Vehicle Routing Problem with Drones
K-NN	K-Nearest Neighbors
LNS	Large Neighborhood Search
LRPD	Location Routing Problem with Drones
MDVRP	Multi-depot Vehicle Routing Problem
MIP	Mixed Integer Program
mTSP	multiple Traveling Salesman Problem
mTSPD	multiple Traveling Salesman Problem with Drones
NNA	Nearest Neighbors Algorithm
PDSTSP	Parallel drone scheduling traveling salesman problem
SA	Simulated Annealing
2EVRP	Two Echelon Vehicle Routing Problem
2EVRPD	Two Echelon Vehicle Routing Problem with Drones
TSP	Traveling Salesman Problem
TSP-D	Traveling Salesman Problem with Drones
VNDS	Variable Neighborhood Decomposition Search
VNS	Variable Neighborhood Search
VRP	Vehicle Routing Problem
VRPD	Vehicle Routing Problem with Drones

ABSTRACT

E-commerce and retail companies are seeking ways to cut delivery time and cost by exploring opportunities to use drones for making last-mile deliveries. In recent years, drone routing and scheduling have become a highly active area of research. This research addresses the concept of a truck-drone combined delivery by allowing autonomous drones to fly from delivery trucks, make deliveries, and fly to delivery trucks nearby. The first part of the research considers the synchronized truck drone routing model by allowing multiple drones to fly from any truck, serve customers and immediately return to any available truck or depot in the system. The goal is to find the optimal routes of both trucks and drones which minimize the arrival time of both trucks and drones at the depot after completing the deliveries. The problem can be solved by the formulated Mixed Integer Programming (MIP) for the small-size problems and our proposed heuristic called Adaptive Insertion Heuristics (ADI) which is based on the insertion technique for the medium/large-size problems. The second part of the research extends the first studied problem by allowing drones to serve multiple customers before merging with trucks as well as considering the capacity requirement for both vehicles. The problem is mathematically formulated and two efficient heuristic algorithms are designed to solve the large-size problems: Drone Truck Route Construction (DTRC) and Large Neighborhood Search (LNS). In the third study, the goal is to study the potential benefits of combining different types of fleet vehicles to deliver packages to the customers. Three types of vehicles are considered in this study including large drones, traditional trucks and hybrid trucks. The problem can be optimally solved by a mathematical formulation on a small scale. Two efficient metaheuristics based on Variable Neighborhood Search (VNS) and Large Neighborhood Search (LNS) are proposed to solve for approximate solutions of the large-size problems. A case study and numerical analysis demonstrate the better delivery time of the proposed model when compared with the delivery time of other delivery models with a single fleet type.

CHAPTER 1. INTRODUCTION

1.1 Drone Delivery from a System Engineering Perspective

1.1.1 The Rise of E-commerce

E-commerce, or electronic commerce, is the buying or selling of goods and services over an electronic network, primarily the internet. E-commerce generates the transactions of the retail products (e.g., clothing/apparel, consumer electronics/technology, and beauty products) between businesses to businesses (B2B), businesses to consumers (B2C) and from consumers to consumers (C2C). These transactions are growing at a very significant rate which has led to reduced in-store sales, and the displacement of brick-and-mortar stores. In most cases, retailers use online and brick-and-mortar stores interchangeably, as they provide multiple options for customers to shop, compare and purchase (omnichannel) (Adams, 2017; Balcik, Beamon, & Smilowitz, 2008).

It is expected that e-commerce will be growing at a significant rate all over the world with the prediction that the global e-commerce sales are likely to reach $4.5 trillion by 2021 from 1.3 trillion in 2014, registering a growth of 246.15 percent ("Global retail e-commerce market size 2014-2021"). Business-to-business sales have represented a larger share of e-commerce but that share is expected to decline by 50% — $1.1 trillion — by 2020 as retail business-to-consumer e-commerce grows. Figure 1.1 represents the prediction of the global e-commerce sales in U.S. dollars (Lee, Chen, Gillai, & Rammohan, 2016).

Because of the increase in the demand for products at e-commerce retailers, logistics and shipping companies have transformed the ways products are delivered to the customers in this highly competitive market. Back then, customers expect to receive a package within 4-6 weeks after purchase, now they expect faster and more flexible delivery options which could come within a week, within one day and even within a same-day delivery. The delivery networks are becoming more localized, shifting the supply chain strategy to focus on regional fulfillment. The goal of this localized approach is to shorten the last mile and remove barriers that stand in the way of making fast deliveries that can be customized to specific times and locations (Hewitt, 2019).

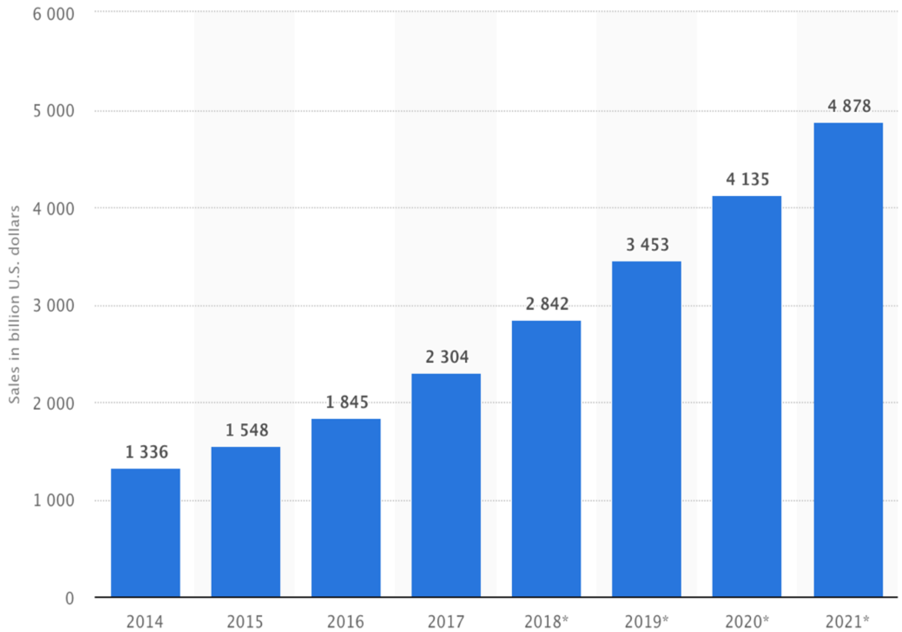


Figure 1.1 Worldwide retail e-commerce sales from year 2014-2021 (Lee, Chen, Gillai, & Rammohan, 2016)

Last-mile delivery is typically the most difficult part of a supply chain and is estimated to account for up to 50% of the total distribution cost. Delivering products in the last-mile stage of the supply chain has been a challenge for e-commerce companies since its inception. To operate the shipping cost-effectively, the logistics carriers have to rely on the two fundamental economics principles: 1) route density — how many packages can be delivered on a given delivery run; and, 2) concentrated package volume — how many packages or items are delivered at each stop. Carriers gain more efficiencies if they can deliver a large number of deliveries per stop and/or per mile, package (Hewitt, 2019).

Major three national carriers (UPS, FedEx, and the USPS) have recently expanded their hubs, delivery networks, extensive fleets to account for 85% of the last-mile market in the U.S. Most deliveries are done by a larger fleet of USPS, UPS and FedEx light and medium trucks with expanded hours of operation to homes and businesses as well as to neighborhood mailbox facilities. The traditional way of making last-mile deliveries requires a delivery person to pick up the packages at a consolidation point and delivers them directly to the recipients. While many of these strategies have increased the productivity of carriers, it comes with a high expense in terms of the labor cost and causes a shortage of truck drivers for all of these deliveries. This has led many tech and e-commerce companies to explore various technologies that will change the way we deliver to meet the needs of the last mile in the future including the use of autonomous vehicles, robots, and even drones. ("E-Commerce and Emerging Logistics Technology Research Report", 2009).

1.1.2 The Future of Last-Mile Delivery

As mentioned in the previous section, technological innovations in the last-mile delivery of products can determine the profitability of companies, both traditional players and new entrants in this growing market. There has been some recent attention on the use of drones, small robots (droids), bike couriers and autonomous ground vehicles (AGVs) with lockers to deliver items to households to optimize the efficiency of the last-mile delivery. (Patil, 2016; Rao et al., 2016).

Autonomous ground vehicles (AGVs) with lockers: AGVs deliver packages without human intervention and customers will be notified for the exact arrival time of the vehicle. The customers are directed to pick up items from the specified locker mounted to the vehicle. AGVs will likely to be used for same-day delivery which requires a fast fulfillment process. The vehicle is expected to save at least 40 percent or more in the cost of operation and can operate 24 hours/day. It can offer overnight pickup and Sunday delivery which offers higher flexibility for the customers to pick up their packages than the current delivery (Joerss et al., 2016).

Small robot (droids): A droid is a small autonomous vehicle that can deliver packages to the doorstep by traveling only on sidewalks and crosswalks at the speed lower than 10 km/h. They must be supervised by humans at the remote distance. They can carry loads as heavy as 50 pounds for as far as 30 miles (Hawkins, 2019).

Bike couriers: Bike couriers are used for instant delivery in urban areas. They are often seen in point-to-point delivery, especially for B2B documents and prepared food. It is considered as the most-cost competitive choice for last-mile delivery. If droids are expensive, bike couriers are likely to be the best instant delivery from point to point (Joerss et al., 2016).

Drones: Drones are autonomous aircraft, e.g., copters or vertically starting planes that can carry small packages (up to 15 kg) to their destination using the most direct route at a relatively high average speed (50-70 mph). They are used in the specific environments, in rural areas, for the specified time window or the same day delivery. At the cost of $1 per shipment, drones are expected to be used by e-commerce companies to expedite B2C delivery which leads to higher revenue and improves customer satisfaction (Desjardins, 2018).

Nevertheless, a traditional B2B delivery still uses a large truck fleet due to a large volume per delivery in one stop. This factor plays an important role in delivery cost as each delivery incurs a huge setup cost and handling cost per stop. Besides, large business customers often require value-adding services, e.g. sorting items into small storerooms which AGVs or drones cannot handle. Thus, shipping couriers will likely to use traditional trucks for large volume B2B shipments with value-adding services and use drones or droids for small package deliveries for B2C (Joerss et al., 2016). Figure 1.2 shows the different types of last-mile delivery in the future.

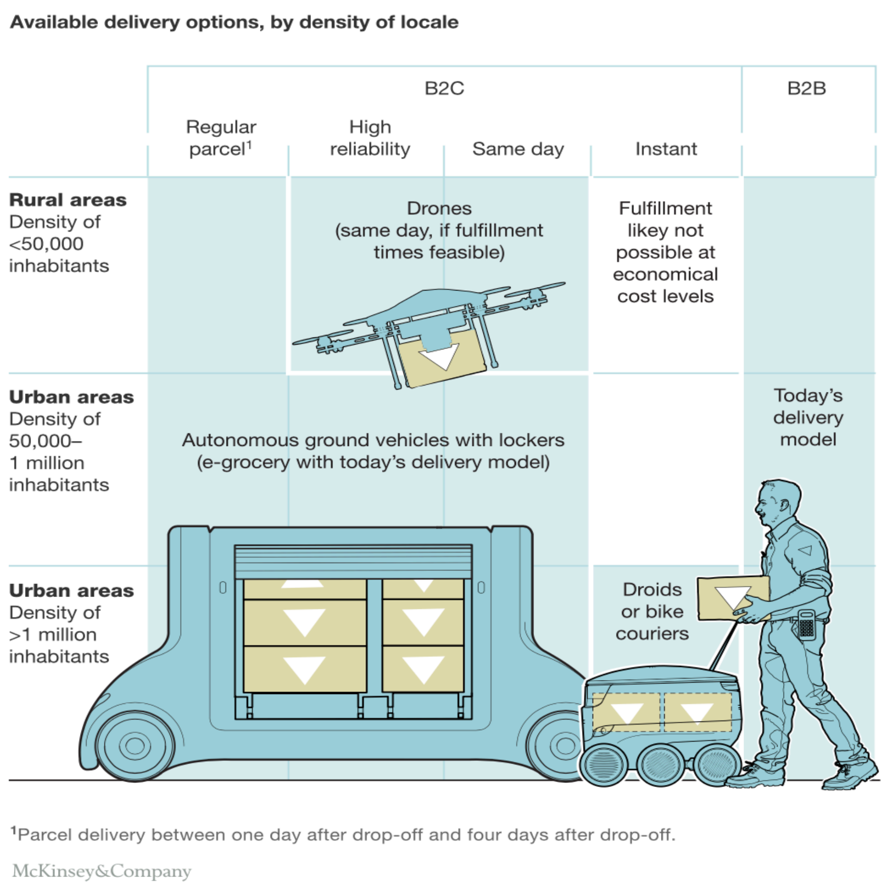


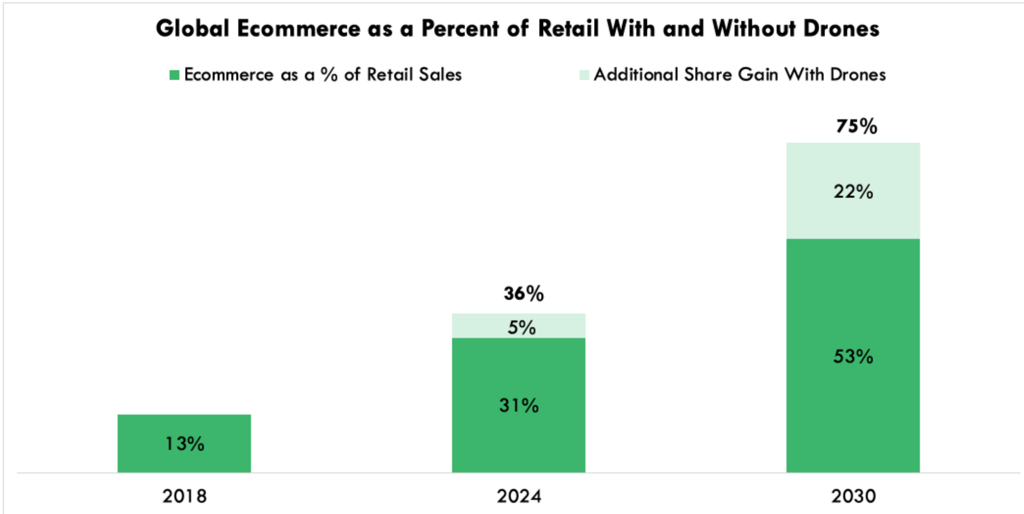
Figure 1.2 Different options for the futuristic last-mile delivery (Joerss et al., 2016)

1.1.3 Drone for Last-Mile Delivery

Given the current growing size of the parcel delivery market, customers are likely to buy products from a company with fast, flexible, and economical parcel delivery. More than three-quarters of online customers would like their orders shipped and delivered the same day (Lindner, Enright, & Bloomberg News, 2017). To fulfill this demand, consumer research has shown that online shoppers are willing to have their purchases delivered by a drone if the drones can deliver faster, more flexible, and cheaper than other vehicles. Major corporations are taking notice and finding new and innovative ways to incorporate drone delivery into their business models. Therefore, the use of delivery drones is on the horizon and their benefits will completely revamp the e-commerce industry ("Drone Delivery: The Future of e-commerce", 2019).

In the past, drones were mostly used for military, surveillance purposes, and a hobbyist's tool for capturing images of foliage, sporting events, and cityscapes. Now, drones are used in commercial applications and just recently grab the market's attention on its delivery application. In the humanitarian sector, drones can be used to transport medicines and relief supplies to affected areas where the road and rail network are severely damaged by the natural disasters. These drones will alleviate congestion on our daily commutes, promote instantaneous shopping and deliver life-saving medical supplies all over the world (Cohn et al., 2016).

ARK Invest illustrates that using drones for last-mile delivery could boost e-commerce's share of retail sales from 13% today to 75% by 2030 as shown in Figure 1.3 (Keeney, 2019). Without drones, e-commerce could account for just over 50% of retail. The company predicts that the revenue from parcel drone delivery could total $400-500 billion by 2030 at a low cost of $1 per shipment. At the moment, parcel delivery revenues are roughly $280 billion today, at roughly $4 per package. As drone deliveries decrease shipping costs, the e-commerce companies can deliver more volumes and generate higher sales due to consumer's preference for quick, inexpensive shipping (Keeney, 2019).



Source: ARK Investment Management LLC, 2019; Emarketer
Note: ARK assumes that drone adoption boosts ecommerce share of total retail by a factor of (1 + the drone adoption rate)

Figure 1.3 Global e-commerce sales with and without drone (Keeney, 2019).

Drone delivery offers potential benefits that revolutionize the last-mile delivery as follows:

1. Speed: Drones can travel "as the crow flies" at the high speed without being stuck in the traffic. Unlike delivery trucks and bike couriers, they can fly in a straight line to their destination.

2. Accessibility: Drones are a great fit for delivering products to the remote area where roads or infrastructure get damaged. They can fly across the water, railroad or even mountains.

3. Cost: Drones are relatively cheaper than other delivery vehicles. They can significantly reduce the human labor costs for delivering packages.

4. Tracking: Drones allow real-time communication and tracking which helps the operators and customers to track where the packages are.

5. Environmental friendliness: Drones use electric batteries as a power source that does not generate massive greenhouse gas like delivery trucks do. They have minimal environmental impact compared with traditional deliveries by trucks or trains.

Amazon, the world's largest e-commerce company, initially brought the concept of drone delivery into public attention in 2013 by announcing the idea of using drones to fulfill the orders by offering the drone delivery services which could deliver customers' orders within 30 minutes through its Prime Air delivery program (Meola, 2017). Following the announcement, many technology companies such as Google, Alibaba and JD as well as traditional logistics providers such as UPS, DHL and FedEx have been experimenting and adopting their drone delivery services, intending to reduce costs and provide cheaper, yet faster and more efficient service (Ungerleider, 2016).

Companies across the world are focusing on drone delivery, On July 2015, a drone delivery company called Flirtey successfully completed the first fully autonomous delivery with a package that included bottled water, emergency food and a first aid kit (Nichols, 2016). Similarly, DHL also has launched a drone delivery service to ship items, such as medication and other urgently needed goods to people located on an island in Germany's North Sea where most traditional delivery options are not available (Hern, 2014). Drones have also been successfully used to deliver food like pizza by Flirtey and Chipotle by Alphabet X, Google (Volkman, 2018). While Amazon made the first public demo of its drone delivery system in the US in March 2017, JD.com, a giant e-commerce company in China, has been aggressively developing its drone delivery service (Vincent, 2017). The company currently has at least seven different types of delivery drones in testing or operation across four provinces in China (Beijing, Sichuan, Shaanxi, and Jiangsu). The drones are capable of delivering packages weighing between 5 and 30 kg (11 to 66 lbs) while flying up to 100 km/hr (62 mph). (see Figure 1.4 for examples of delivery drones).

a.) b.)

Figure 1.4 Examples of delivery drones from JD and Amazon. a.) Different types of JD drones (Vincent, 2017), b.) Amazon's Prime Air drone (Meola, 2017).

1.2 Regulation

Although using drones for package delivery has recently been tested by many companies in the U.S., the government still prohibits the actual use of drones in the commercial purpose due to its emerging regulation. Modern regulation of commercial drone deliveries is still evolving, and companies cannot experiment with their drone deliveries in the U.S. unless permitted by the FAA (Dorr & Duquette, 2016) .The FAA's regulatory compliance requires the drone operators to conduct the flight within a line-of-sight, keep the drones under 400 feet, register each drone, and not operate the drones within the restricted population-based location or near the airport (Romm, 2017). Due to this, e-commerce and logistics companies have been testing their drone delivery programs in Canada, the U.K., and the Netherlands, which have fewer restrictions than in the U.S. (Bonnington, 2017). The drone itself also has some restrictions on payload and the design to comply e.g. maintaining the payload less than 55 pounds. On the other hand, the Civil Aviation Administration of China (CAAC) permitted JD.com and SF Holding Co., the country's biggest express-delivery company, to start sending packages by drone in certain rural areas. Governments in other countries including U.S. are also developing the regulation on commercial drones to address the low-level air-traffic system to prevent drones from hitting each other or traditional aircraft ("China Is on the Fast Track to Drone Deliveries", 2018).

Regarding air transportation management, the FAA has recently partnered with NASA to form an Unmanned Aircraft Systems (UAS) traffic management platform, essentially an air traffic control system for drones. To manage safety, drone manufacturers and software providers are quickly developing technologies like geo-fencing and collision avoidance to ensure safety when drones fly autonomously. The increase in drone technology development is pushing governments to create new regulations that balance safety and innovation (Meola, 2016). In the most recent update, the White House announced the launch of a three-year pilot program to create “innovation zones” to evaluate commercial drone operations flying at night, flying over people, and flying outside a pilot’s line of sight (McFarland, 201). This program allows logistics companies to test their drone delivery service without any restrictions. With support from different public sectors to finalize the regulation and air transportation platform, commercial drone delivery in the U.S. should be ready to be implemented soon.

1.3 Research Motivation

Compared to traditional delivery modes, drones seem to be a potential fit for the last-mile delivery due to their high travel speed and ability to access areas where no other mode of transportation is available or in areas of high delivery density to augment the capability of the driver. Most drones developed by Google, Amazon, DHL, and JD fly at a speed of 30–62 miles (48-100 kilometers) per hour with a flight range of 10–62 miles (16-100 kilometers) (Heath, 2015). Since drones are not restricted by road infrastructure, they can deliver packages faster than a truck from the same location. While trucks must travel on streets and avenues using rectilinear distance, drones can travel through areas like mountains, jungles, and rivers with relative ease, and take shorter routes (Bamburry, 2015; Berg, 1991; Goel & Kok, 2012). However, drones' disadvantages are mainly their small shipping capacity and their battery run time; e.g., the Amazon Prime drone can carry approximately 5 pounds and their batteries must be swapped or recharged after 30 minutes (Brar et al., 2015). Please note that the recent development in drones makes it possible for drones to carry the package weighing between 5 to 30 kg (11 to 66 lbs) while flying up to 100 km/hr (62 mph) (R., 2017).

To utilize the truck's capacity and drone's delivery speed, Murray and Chu (2015) proposed the new routing model in which a truck and a drone works together as a synchronized working unit to make a delivery. A single drone is attached to a truck where the drone is launched from the truck, serves the customer, and returns to the truck at another location, where the drone gets its battery serviced at the truck. The authors name the model "Flying Side Kick Traveling Salesman Problem" (FSTSP), which aims to minimize the total time of the tour by combing truck and drone for making deliveries. This problem is one of the first representative problems and integrates drones into the classical Traveling Salesman Problem (TSP). Figure 1.5 represents the feasible tour constructed by truck alone and a synchronized truck and drone vehicle.

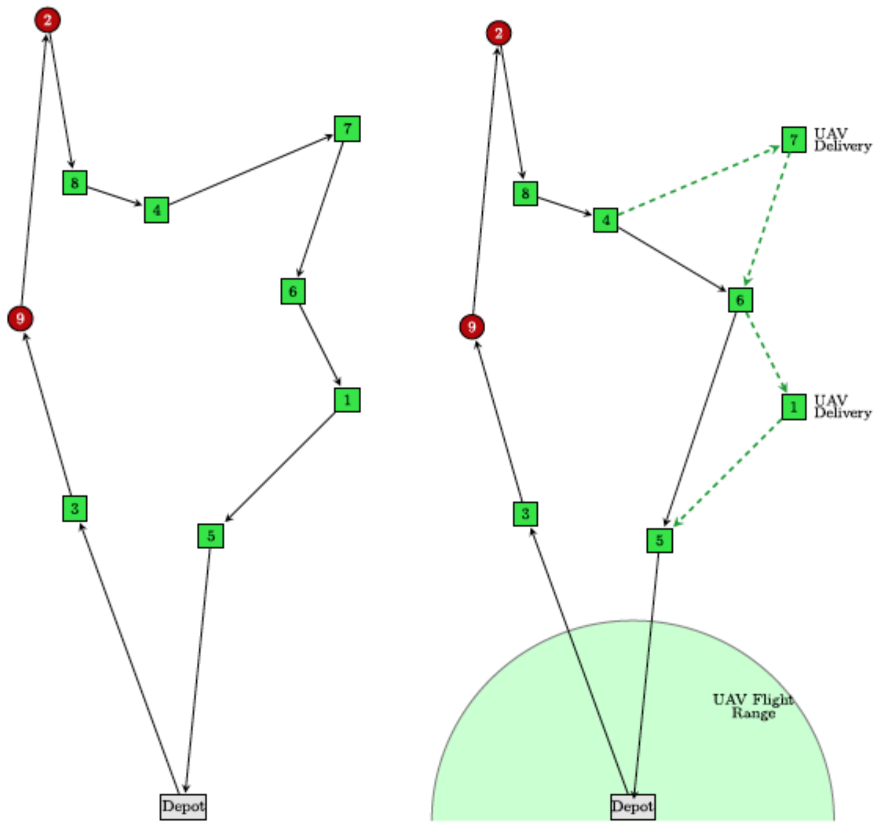


Figure 1.5 The solution tour of truck alone delivery and synchronized truck-drone delivery

Although the model shows promise to reduce costs and delivery times compared to the traditional delivery, it only covers the routing operation which includes simply one truck and one drone in the setting. Inspired by Murray and Chu's proposed model, we want to examine this problem more in detail to further develop the variations of synchronized truck-drone routing models. These models could potentially include multiple fleets of both vehicles with capacities and other side constraints which represent a more realistic scenario of the distribution. This study aims to explore ways to improve the cost and time efficiency of truck-drone hybrid delivery systems. We believe that the application of the proposed models will increase the efficiency and effectiveness of using truck and drone combing operation in last-mile delivery operations.

1.4 Organization of the Dissertation

The remainder of the dissertation is organized as follows. Chapter 2 reviews related literature in (1) Drone Routing Problem; (2) Multiple Traveling Salesman Problem; (3) Vehicle Routing Problem, and (4) Two Echelon Vehicle Routing Problem. Chapter 3 presents the Multiple Traveling Salesman Problem with Drones with mathematical formulation, Adaptive Insertion Heuristics and experimental results. Chapter 4 presents the Two Echelon Vehicle Routing Problem with Drones followed by the mathematical formulation; two heuristics approaches: Drone Truck Route Construction and Large Neighborhood Search; and computational results. The Integrated Vehicle Routing Problem with Drones is presented in Chapter 5 with the mathematical formulation and computational results on benchmark problems and a case study. Lastly, Chapter 6 concludes the dissertation and presents directions for future research.

CHAPTER 2. BACKGROUND AND LITERATURE REVIEW

2.1 Drone Routing Problem

This section reviews related literature on the well-studied drone routing problem. Due to the rapidly growing popularity, many works related to the drone routing optimization problems are found in the literature. This section presents the most relevant papers in this area, which can be categorized into two variants of the classical routing models: the Traveling Salesman Problem (TSP), in which only one truck is employed, and the Vehicle Routing Problem (VRP), in which multiple trucks are employed. A summary of the papers is given in Table 1.1.

2.1.1 Drone Routing as an Extension of TSP

We are aware of many related works in drone routing problems. In 2015, Murray and Chu (2015) introduced a new type of TSP problem called the "Flying Sidekick Traveling Salesman Problem" (FSTSP). The authors proposed the idea of having a drone attached to the top of a truck that can make a delivery while the truck is making another delivery simultaneously. Once the drone finishes making a delivery, it needs to fly back to the truck at the current delivery location or along its route to the next delivery location. FSTSP considers only one truck and one drone. The model is inspired by the Vehicle Routing Problem (VRP) with synchronization constraints (Drexl, 2012). The same authors also proposed another model called the "Parallel Drone Scheduling Traveling Salesman Problem" (PDSTSP) in which trucks and drones work independently to serve all customers.

Besides FSTSP, there are well studied drone-truck routing problems in the past literature with similar features but can be solved by different efficient approaches. Agatz et al. (2018) studied a similar problem called the "Traveling Salesman Problem with Drone" (TSP-D), in which a truck and a drone are making deliveries in parallel. The authors presented the new MIP model and provided efficient heuristics based on local search and dynamic programming. Bouman et al. (2018) recently developed an exact solution for the TSP-D based on dynamic programming. Their approach can successfully solve larger problems than with the mathematical programming approaches by Agatz et al (2018). The authors highlighted that their solutions can be improved by adding the additional constraint that restricts the number of drone sorties. Ponza (2015) examined the FSTSP in detail and applied the simulated annealing technique to search for good solutions. Ha et al. (2018) proposed the min-cost TSP-D with the objective to minimize the total transportation cost. This work included the model and the two algorithms: TSP-LS based on local search and a Greedy Randomized Adaptive Search Procedure (GRASP) to solve a new TSP-D problem. Yurek and Ozmutlu (2018) provided an iterative optimization algorithm for the TSP-D based on the decomposition of the problem into two components: (1) finding a truck route and (2) finding the optimal drone routes within the truck route by solving a Mixed-Integer Linear Programming. Their approaches were tested and showed an improvement from the results from Murray and Chu (2015), and Agatz et al. (2018). Marinelli et al. (2017) extended the TSP-D by allowing a drone to be launched and merge with a truck at any location along a route arc (en-route). The authors proposed a novel heuristic based on a greedy randomized adaptive search procedure (GRASP) to solve benchmark instances. Jeong et al. (2019) modified the FSTSP to consider the effect of the payload on the UAV energy consumption and restricted flying areas.

In a similar theme, Ferrandez et al. (2016) proposed an optimization model of a truck-drone system in tandem delivery networks by using the K-means algorithms to find the most efficient launch locations as well as using a genetic algorithm to assign the truck route between those launch locations. Mathew et al. (2015) proposed a new drone and truck problem called the "Heterogeneous Delivery Problem" (HDP), which shares similar features with TSP-D and FSTSP. The goal is to seek an optimal delivery route in an urban location with the objective to minimize the total cost of deliveries, which consists of the cost of truck travels, the cost of drone travels, and the cost of simultaneous trucks and drones travels. The authors obtained solutions by converting HDP to "Generalized Traveling Salesman Problem" (GTSP). They also proposed another model "Multiple Warehouse Delivery Problem" (MWDP), a variant to the HDP which drones serve customers from multiple warehouses. Kim and Moon (2019) developed the "TSP with a drone station" (TSP-DS), which has similar features to the PDSTSP but includes a drone station as the facility where drones and charging devices are stored. Tu et al. (2018) proposed the new problem as an extension of the TSP-D problem in which a truck travels with multiple drones called "TSP-mD."

Moshref-Javadi and Lee (2017) proposed the Traveling Repairman Problem with Drones (TRPD) with two objectives: 1.) to minimize the sum of latencies at customers and 2.) to minimize the largest latency (the waiting time of the last visited customer). At each stop point, the trucks must wait for drones after they are launched to make deliveries. The Mixed-Integer Programming formulation was presented together with a worst-case analysis of the problem. Kitjacharoenchai et al. (2019) proposed the "multiple Traveling Salesman Problem with Drones" (mTSPD), which has the same operation as FSTSP but utilizes multiple trucks and drones and allows a drone to be retrieved by any truck that is nearby and not necessarily the same truck that it is launched from. Murray and Raj (2019) introduced the "Multiple Flying Sidekicks Traveling Salesman Problem" (mFSTSP), which is an extension of their previous work (FSTSP) with the consideration of an arbitrary number of heterogeneous UAVs that may be deployed from the depot or the delivery truck. The authors provided the Mixed Integer Linear Programming (MILP) formulation along with the three-phased heuristic solution approach.

2.1.2 Drone Routing as an Extension of VRP

In a recent work, Campbell et al. (2017) proposed continuous approximation (CA) models to obtain the optimal number of truck and drone deliveries per route, the optimal number of drones per truck and the total cost of operation in the hybrid truck-drone delivery problem. Similarly, Carlsson and Song (2018) implemented a CA technique to determine the best set of parameters that results in the minimum completion of all truck-drone deliveries in the Euclidean plane. The proposed method was tested on the generated instances (with up to 500 vertices) and practical data (with up to 100 vertices) to support their hypothesis. In this problem, the customers are located within the drone’ flight range near the depot. Drones serve customers close to the depot while trucks serve the rest of the customers that might be located far away from the depot. Trucks and drones work independently to serve all customers.

Dorling et al. (2017) in another perspective, proposed the VRP based models: One on the total delivery costs subject to a delivery time limit and another on the overall delivery time subject to a budget constraint. The author also provided the mathematical model for a cost function that considers energy consumption and drone reuse. Wang et al. (2016) provided worst-case analyses for the “Vehicle Routing Problem with Drones” (VRPD) problem by studying how the two parameters—the number of drones per truck and the speed of the drones—can affect the maximum savings from using drones. Poikonen et al. (2017) considered the same work by finding upper bounds on the amount of time saved from drone operations with an integration of the drone's battery life, cost metrics, and fixed cost of deploying drones. The same authors further continued their work of VRPD by expanding their previous results to fit with the wide variety of circumstance and establish the bounds that relate to the Vehicle Routing Problem (VRP) and the "min-max Close Enough Traveling Salesman Problem" (CETSP) (Poikonen, Wang, & Golden, 2017). Schermer et al. (2018) formulated a Mixed-Integer Linear Program for VRPD, which is considered as a variant of VRP. The MILP can be solved via any commercial solver in the small-size problem. The authors proposed an algorithm based on the well-known Variable Neighborhood Search (VNS) approach.

Pugliese and Guerriero (2017) introduced the "Vehicle Routing Problem with Drones and Time Windows" (VRPDTW). The authors provided a mathematical model for the VRPDTW which can be used on the commercial solver for 5-10 nodes problem size. Ulmer and Thomas (2017) proposed the "Same-day Delivery Routing Problems with Heterogeneous Fleets" (SDDPHF) and modeled it as a Markov decision process. The model is considered as a relaxed version of the PDSTSP of Murray and Chu (2015) in which all customers are known in advance. Cheng et al. (2018) proposed a "Multi-Trip Drone Routing Problem" (MTDRP) in the drone-only system in which the drone can visit multiple customers per trip. The authors considered the influence of payload and distance on flight duration. They provided two formulations and the exact algorithms for the model. Dayarian et al. (2020) presented a "Vehicle Routing Problem with Drone Resupply" (VRPDR) in which a fleet of drones and a fleet of vehicles collaboratively perform home deliveries of online orders from a fulfillment center. Two heuristic approaches were developed for the special case in which the system consists of a single drone and a single delivery vehicle. Ham (2018) extended the PDSTSP by considering two different types of drone tasks: drop and pickup. A constraint programming method was proposed for different numbers of trucks, drones and depots. Hong et al. (2017) developed a heuristic model to obtain the optimal location of drone recharging stations by connecting the stations and delivery locations based on a continuous space shortest path.

Luo et al. (2017) proposed a two-echelon cooperated routing problem for the ground vehicle (GV) and its carried unmanned aerial vehicle (UAV) (2E-GU-RP). The problem is very similar to VRPD proposed by Schermer et al. (2018) but allows drones to make multiple deliveries in one trip. Karak and Abdelghany (2019) presented the "Hybrid Vehicle-Drone Routing Problem" (HVDRP) for pick-up and delivery services in which multiple drones can be dispatched from a mothership to perform dozens of pick-ups and deliveries simultaneously. Wang and Sheu (2019) presented the "Vehicle Routing Problem with Drones" (VRPD) with a distinctive feature that allows drones to make multiple deliveries per trip and return to any available truck in the fleet. The authors proposed a Mixed-Integer Programming model and developed a branch-and-price algorithm to solve VRPD for the exact solution. Poikonen and Golden (2020) recently developed the "k-Multi-visit Drone Routing Problem" (k-MVDRP), which considers a tandem between a truck and k drones allowing a drone to deliver one or more packages to customers. Lastly, Kitjacharoenchai et al. (2020) proposed a "Two echelon vehicle routing problem with drones in last-mile delivery" that addresses two levels (echelons) of delivery: primary truck routing from the main depot to serve assigned customers and secondary drone routing from the truck to serve other sets of customers.

Table 1.1 Summary of papers on drone routing problems

Model	Reference	Truck-Drone Synchroniz	Routing characteristics						
			Vehicle		Capacity		Drone trip		Combining different/hete rogenrous vehicle fleets
			# Truck	# Drone	Truck	Drone	# Multiple deliveries per trip	Return to different	
FSTSP	Murray and Chu (2015)	X	Single	Single	-	-	-	-	-
TSP-D	Agatz et al. (2018)	X	Single	Single	-	-	-	-	-
mFSTSP	Murrary and Raj (2019)	X	Single	Multiple	-	-	-	-	X
TSP-mD	Tu et al. (2018)	X	Single	Multiple	-	-	-	-	-
Min-Cost TSPD	Ha et al. (2018)	X	Single	Single	-	-	-	-	-
HDP	Mathew and Smith (2015)	X	Single	Single	-	-	-	-	-
TSP-D /FSTSP	Campbell et al. (2017)	X	Single	Multiple	-	X	-	-	-
TSP-D	Ferrandez et al. (2016)	X	Single	Multiple	-	-	-	-	-
MC-DDP	Dorling et al (2017)	-	N/A	Multiple	-	X	X	-	-
TRPD	Moshref-Javadi and Lee (2017)	X	Single	Multiple	-	-	-	-	-
TSP-DS	Kim and Moon (2019)	X	Single	Multiple	-	-	-	-	-
2E-GU-RP	Luo (2017)	X	Single	Single	-	-	X	-	-
VDRPTW	Pugliese and Guerriero (2017)	X	Multiple	Multiple	-	-	-	-	-
mTSPD*	Kitjacharoenchai et al (2019)	X	Multiple	Multiple	-	-	-	X	-
SDDPHF	Ulmer and Thomas (2017)	-	Multiple	Multiple	-	-	-	-	-
VRPDR	Dayarian et al. (2017)	X	Single	Single	-	X	X	-	-
MTDRP	Cheng et al. (2018)	-	N/A	Multiple	-	X	X	-	-
VRPD	Schermer et al. (2018)	X	Multiple	Multiple	-	-	-	-	-
VRPD	Wang et al. (2017), Poikonen et al	X	Multiple	Multiple	X	-	-	-	-
PDSTSP	Ham (2018)	-	Multiple	Multiple	-	-	X	-	-
2EVRPD**	Kitjaharoenchai et al (2020)	X	Multiple	Multiple	X	X	X	-	-
HVDRP	Karak and Abdelghany (2019)	X	Multiple	Multiple	-	X	X	X	-
VRPD	Wang and Sheu (2019)	X	Multiple	Multiple	X	X	X	X	-
k-MVDRP	Poikonen and Golden (2020)	X	Single	Multiple	-	-	X	-	-
I-VRPD***	Kitjaharoenchai et al (Working)	X	Multiple	Multiple	X	X	X	-	X

2.2 Multiple Traveling Salesman Problem

In this section, we provide the background on the multiple Traveling Salesman Problem (mTSP) which is closely related to our multiple Traveling Salesman Problem with drones (mTSPD) in Chapter 3. The mTSP is a generalization of the well-known TSP. The Travelling Salesman Problem (TSP) aims to determine a set of routes for a salesman TSP to visit *n* cities given that a salesman must visit each and every city exactly once and finally comes to the initial position with the objective to minimize the total cost of visiting all nodes (Dantzig et al. ,1954). In mTSP, routes are determined for *m* salesmen instead of one. Since the TSP is a special case of the mTSP, all the formulations and algorithms developed in the literature for the TSP are valid for the mTSP and vice versa. The mTSP is considered as an NP-Hard like TSP and VRP and requires an efficient heuristic approach to obtain the solutions once the problem size gets big. Several modern heuristics have been proposed to solve the mTSP including but not limited to an artificial neural network (NN), genetic algorithms (GA) and simulated annealing (SA) (Bektas, 2006).

The mTSP can be considered as a relaxation of the famous Vehicle Routing Problem with the capacity constraints and demand associated with each customer removed. Because of this, all the formulations and algorithms proposed for the VRP are also valid and applicable to the mTSP, by assigning sufficiently large capacities to the salesmen (vehicles) (Bektas,2006). The mTSP can be used in a various planning/scheduling applications, ranging from machine scheduling of prints to routing in transportation. Figure 2.1 represents the illustration of TSP, mTSP and VRP solution.

2.3 Vehicle Routing Problem

The Vehicle Routing Problem (VRP) is one of the most well-known hard combinatorial problems which aims to design an optimal route for a fleet of vehicles to service a set of customers, subject to different sets of constraints. It considers the problem of routing vehicles from a central depot to serve customers with known demands with the goal to minimize the total travel cost. The VRP is a generalization of the multiple Traveling Salesman Problem (mTSP) that has many practical applications (Kuma, 2012). The vehicle routing problem (VRP) has been very extensively studied in the optimization literature. It was first introduced in 1959 by Dantzig and Ramser (1959) as a generalization of the TSP and mTSP. The VRP is an NP-hard problem that can be exactly solved only for small-size instances. Recent research in VRP primarily focuses on the development of heuristic and metaheuristic approaches to obtain good quality solutions in the limited time for the real-world problems that are characterized by large vehicle fleets and the number of customers (Kuma, 2012; Laporte, 2009). The VRP can be categorized into 1.) deterministic version which corresponds to the case where the number of nodes, the number of vehicles, the demand of each node, and the travel time between each node, etc. are known in advance; and 2.) stochastic version which includes probability and uncertainty on the input data (Yalcindag et al., 2011).

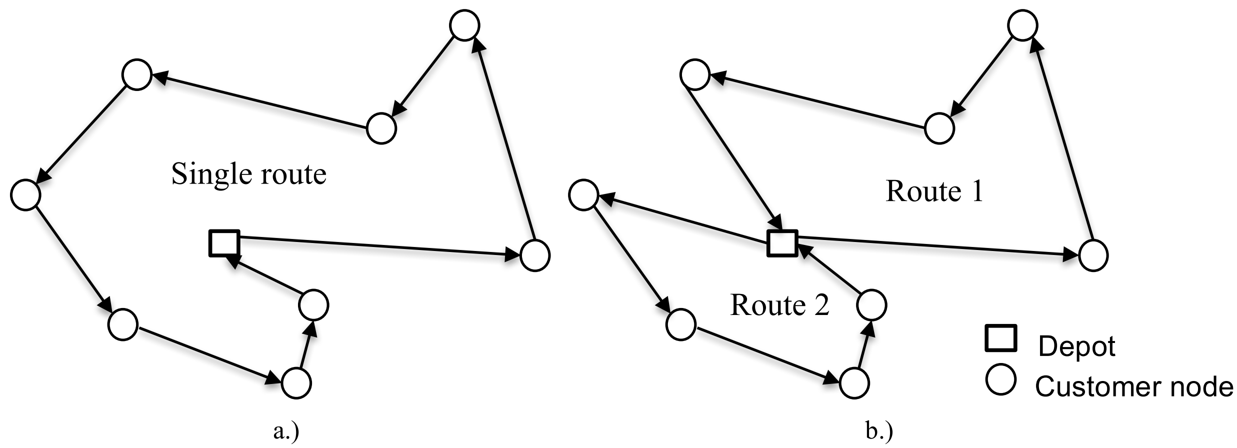


Figure 2.1 Illustration of the a.) Traveling Salesman Problem (TSP) and b.) Multiple Traveling Salesman Problem (mTSP) / Vehicle Route Problem (VRP) route solutions.

The most common variants of the VRP include:

1. Capacitated VRP (CVRP): The problem takes into account vehicle capacity and nodes are assigned with specific demands.

2. VRP with Multiple Depots (MDVRP): The problem requires the assignment of customers to depots. A fleet of vehicles is based at each depot. Each vehicle departs from the specific depot, serves all customers assigned to that depot, and returns to the same depot. The VRP with a single depot is considered as CVRP which all vehicles start and end their routes at a single depot in a single depot VRP.

3. Open VRP (OVRP): The CVRP with the term that a vehicle does not have to return to the depot it departs from.

4. VRP with Distance Constraints (DCVRP): The capacity constraint is replaced by a maximum length (time) constraint of the route.

5. VRP with Backhauls (VRPB): The vehicle must first serve the set of customers with a certain amount of products from the depot. After completing the delivery phase for all customers in the first set, the vehicle must pick up the collection of a certain amount of products from the customers in the second set and return to the depot.

6. VRP with Picks and Deliveries (VRPPD): A number of goods need to be transported from certain pickup locations to other delivery locations.

7. VRP with Time Windows (VRPTW): Each customer must be served within a certain time window interval.

There are many more variants of the VRP with a variety of constraints. For more detail on different versions of the VRP, please refer to the book of Toth and Vigo (2002). Figure 2.2 represents the between TSP, mTSP, VRP, and their variants (Yalcindag et al., 2011)

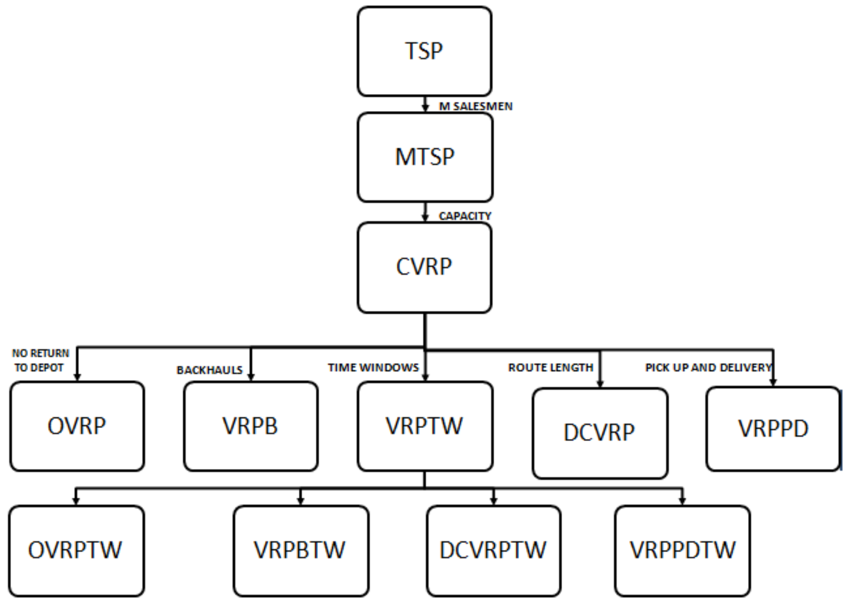


Figure 2.2 Relationship diagram between different routing problems (Yalcindag et al., 2011).

The VRP can be solved by the exact algorithms which consist of

1.) Branch-and-Bound
2.) Dynamic Programming
3.) Vehicle Flow Formulations
4.) Commodity Flow Formulations
5.) Set Partitioning Formulations (Laporte, 2009).

Between 1964 and the early 1990s, many classical heuristics were developed to solve VRP which include 1.) Savings Algorithm, 2.) Set Partitioning Heuristics, 3.) Cluster-First, Route-Second Heuristics, and 4.) Improvement Heuristics (Toth and Vigo, 2002). In recent years, metaheuristics have primarily been used to obtain a sufficiently good solution. These well-known metaheuristics include 1.) Local Search such as tabu search, variable neighborhood search, and adaptive large neighborhood search; 2.) Population Search such as genetic algorithm (GA) and Ant colony optimization; and 3.) Neural networks (Toth and Vigo, 2002; Laporte, 2009).

2.4 Two Echelon Vehicle Routing Problem

Two Echelon Vehicle Routing Problem was introduced by Crainic et al. (2007) as a new problem class to solve the routing operation for the city logistics system. It is considered as a variant of the VRP, which considers two distribution levels: trucks operate on the first level between a central depot and selected intermediate depot, called satellites. The second level considers the distribution between the satellites and the end customers (Breunig et al, 2015). Cuda et al. (2015) defines the Two-Echelon Vehicle Routing Problems (2EVRP) as the problem which involves only tactical planning decisions, and the routing is present at both echelons. This is a two-stage routing problem that we solve for the routes from depot to satellites and develop *n* routings for *n* satellites.

The objective of 2EVRP is to serve customers by minimizing the total travel cost and satisfying the capacity constraints of vehicles. There is a single depot and a fixed number of capacitated satellites. All customer demands are fixed and known in advance (Feliu et al., 2008). The 2EVRP is a generalization of the classical VRP and is considered as an NP-hard. It integrates two levels of VRP in which both levels have to be synchronized. The first level of the 2EVRP reduces to a CVRP with split deliveries whereas the structure of the second level is a multi-depot Vehicle Routing Problem (MDVRP) (Breunig et al., 2015). Figure 2.3 illustrates the solution of the 2EVRP routing problem.

The 2EVRP has received a lot of attention in the recent decade as researchers are looking for the solution approaches to solve this problem effectively. Crainic et al. (2008) developed two heuristics for 2EVRP based on a two-phase approach where the first and the second echelon routing problems are separated and solved sequentially. The same authors later proposed a multi-start heuristic to solve 2EVRP by first assigning customers to satellites heuristically and solving the VRP with the exact method (Crainic et al., 2011). Jepsen et al. (2012) presented a branch-and-cut method to obtain an optimal solution for some of the 2EVRP instances. Santos et al. (2014) presented a branch-and-cut algorithm and reported the optimal solutions up to 50 customers. The current state-of-the-art heuristic for the 2EVRP is the ALNS algorithm introduced by Hemmelmayr et al. (2012), which implements the destroy and repair operators iteratively. Lastly, Breunig et al. (2016) proposed a large neighborhood-based heuristic for 2EVRP which combines enumerative local searches with destroy-and-repair principles.

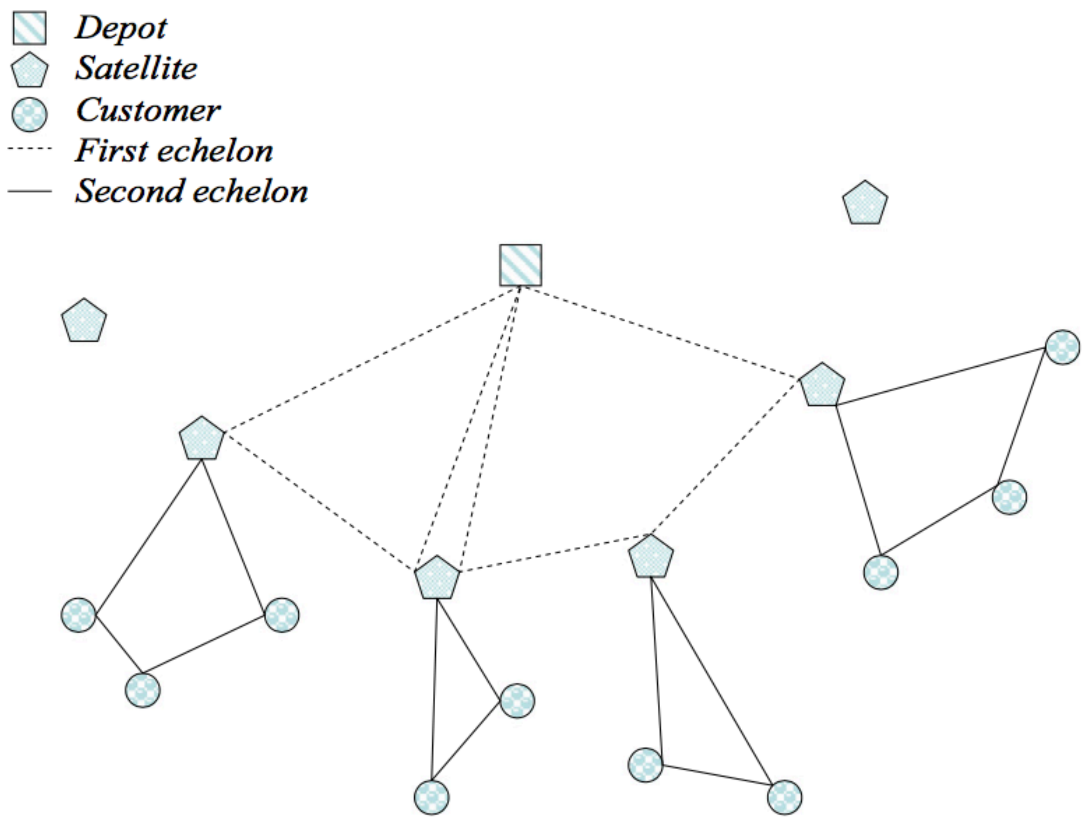


Figure 2.3 Illustration of Two Echelon Vehicle Routing Problem (2EVRP) route solution (Cuda et al., 2015).

CHAPTER 3. MULTIPLE TRAVELING SALESMAN PROBLEM WITH DRONES

3.1 Problem Description

The multiple Traveling Salesman Problem with Drones (mTSPD) is an extension of the multiple Traveling Salesman Problem (mTSP) with the implementation of drones in the operations. The mTSPD model formulation is derived from the FSTSP model in Murray and Chu (2015) with some additional constraints from the mTSP. In the mTSP, the *m* salesmen must visit *n* nodes, forming totally *m* tours, one per salesperson. There are two main sets of constraints in the mTSP problem. The first set of constraints requires that all salesmen must depart and return to the starting node (depot) at the end of the trip, no matter which tour they choose. The second set of constraints states that every salesman must travel to a specific set of customers between the first and the last node given that each node can only be visited once by the assigned salesman except the starting node (depot) (Bektas, 2006). The typical objective of the mTSP is to find the total shortest tour that each salesman must travel from the depot to visit the assigned set of cities and back to the depot. In the mTSPD, the objective is to minimize the time that the last delivery is completed similar to min-max mTSP, which aims to minimize the maximum tour length (time) of each salesman with the purpose to divide the cost (time) of tours among each salesman equally (Bertazzi, Golden, & Wang, 2015; Kivelevitch, Cohen, & Kumar, 2013). The solution of mTSPD requires exactly *m* tours, which is equal to the number of salesmen.

In this problem, a truck is treated as a salesman because we do not consider the capacity of trucks and demand quantity of customers. We assume that each truck has a sufficiently large capacity to carry both packages and drones through the entire operation. At the depot, the departure times of all trucks are zero. All trucks must initially depart from the depot, serve all customers, and return to the depot. The model will determine the set of customers specifically visited by trucks and the set of customers visited by drones. It is required that each customer must be visited exactly once by either a truck or a drone.

The mTSPD model represents the actual operation similar to the truck-drone delivery test conducted by UPS in February 2017 (Hughes, 2017; Zito & Radocaj, 2016), which showed the case when a drone launched from atop one of the UPS delivery trucks autonomously delivers a package and then returns to the vehicle. This happens simultaneously while a driver continues to drive along a route, delivering packages. In addition, we assume the drone can be retrieved by any truck that is nearby and not necessarily the same truck that it is launched from. Figure 3.1 represents an illustrative problem and different types of solutions from mTSPD comparing to the mTSP and TSP on the same problem. The solid line in all Figures 3.1a.) to 3.1f.) refer to the truck tour and the dotted line refers to the drone tour.

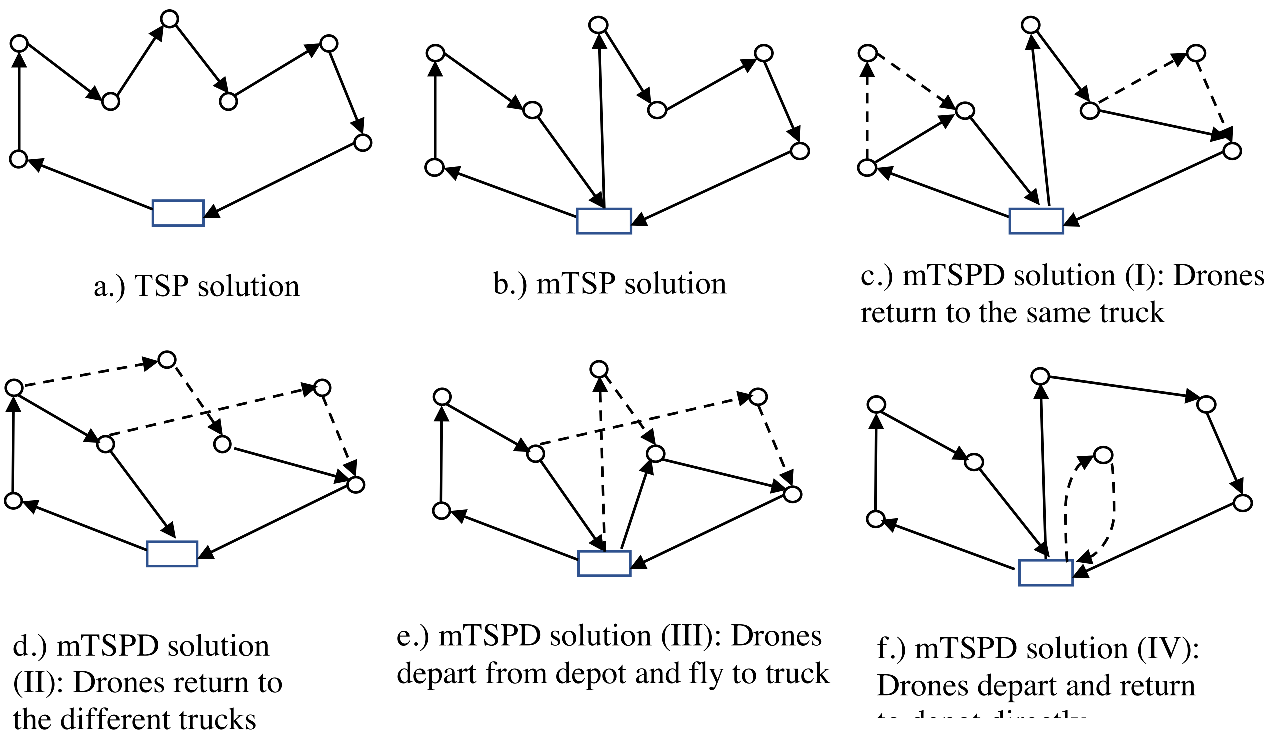


Figure 3.1 Illustration of the feasible solutions from TSP, mTSP and mTSPD.

3.2 Assumptions and Contributions

3.2.1 Assumptions

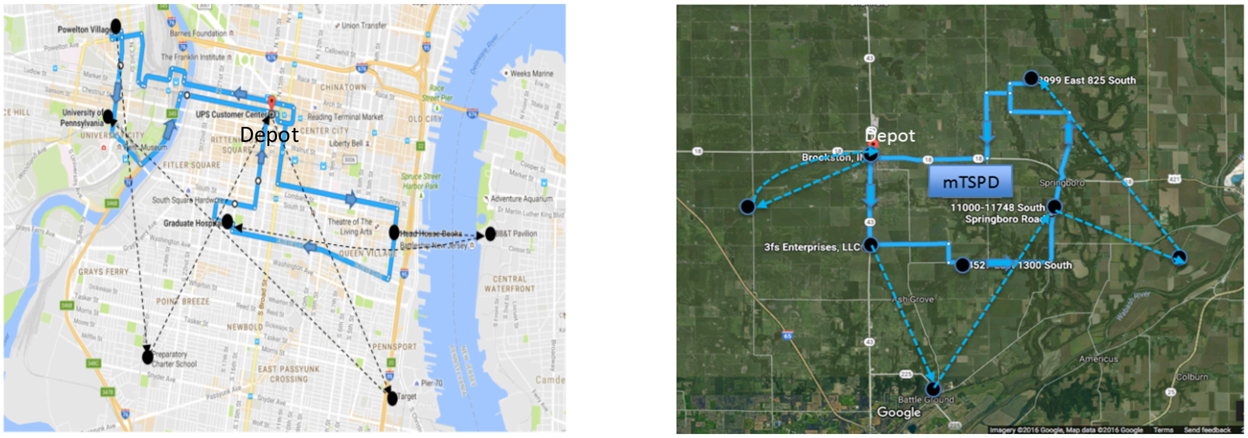
Turning from the described operations to the routing problem, several assumptions must be made:

- We disregard the capacity of each truck and assume each truck has a sufficiently large capacity to carry both packages and drones through the entire operation. This is to reduce the complexity of the problem and to ensure that a truck can operate without exceeding its capacity.
- Drones are assumed to be homogeneous with the same configuration and can carry one package with a small payload up to 5 pounds (2.3 kg) at a time (Glaser, 2017). Once the drone finishes the delivery to a customer, it must immediately fly to any customer node that has not been visited yet, or it can fly back to the depot directly. Due to these characteristics, drones can be launched from a truck and return to another truck in the fleet. There are no specific assignments of drones to trucks once they leave the depot.
- In our model, we disregard the set-up and recovery times when the drone is launched or retrieved at a particular node. This set-up and recovery time can be negligible since their values are small compared to the truck and drone travel time and will not affect how the solution route is determined in the model. We also assume that the drone can complete its delivery and return to the truck before it runs out of battery, or it can land on a recharging stations to extend its flying range (Hong, Kuby, & Murray, 2017; Gentry, Hsieh & Nguyen, 2016).
- Drones can only merge with a truck at a customer node and are not allowed to merge with a truck in any intermediate location. From a practical view, it would be difficult for a drone to land on a moving truck, as both of their speeds have to match (Buchmueller, Green, Kalyan, & Kimchi, 2016). Merging a drone with a movable truck requires the drone to reduce its speed and the truck to increase its speed, which violates the constant speed assumption of the drone and the truck and would decrease a drone's performance from flying at full speed. Also, trucks and drones must wait for each other whenever one arrives at the customer node before the other. This assumption is to ensure that the drone is serviced prior to departure and right after merging with the

truck. Once the drone arrives at the truck, it is serviced for the functional check and battery charge at the truck.

- Multiple drones can fly simultaneously; however, only one drone can be launched or retrieved at each particular node at the same time. In other words, the truck cannot launch or retrieve more than one drone in any customer node. Allowing multiple launches and retrievals requires that the delivery truck must reserve a certain amount of space for keeping drones. Currently, our model does not include the capacity constraint of the drone's space which refers to the truck's capacity to carry some finite number of drones. Without limiting the number of retrievals, one of the feasible solutions is to launch as many drones as possible from a single node. For operational simplicity and to reduce model complexity, it would be safe to limit the number of launches and retrievals to just one.
- We assume that the FAA regulations about the visual line-of-sight (VLOS) can be relaxed by allowing a first-person-view piloting. In this system, the drone pilot controls the drone, as he can see what the drone sees as it flies (Browne, 2017).

Figure 3.2 represents the result of two generated case studies we have conducted using the mTSPD model to generate the solution routes. The blue solid lines indicate the truck's path and that dashed lines indicate the flight path of the drone. Figure 3.2a.) shows the paths of both trucks and drones. These case studies illustrate the advantages of drones compared with the trucks in delivery operations. The trucks can make deliveries to multiple customers on the tour while the drone can deliver the package to a specific customer at a faster speed. It also shows the substantial advantage of the drone, as it does not get stuck in traffic and can travel across the river quite easily. Figure 3.2b.), on the right, similarly shows that the drone can fly across areas where the road has no access to make a delivery. In both cases, the drone gains the advantage of flying in a Euclidean travel space while the truck has to follow the road distance.



a.) b.)

Figure 3.2 Examples of mTSPD solution routes from case studies. a.) The solution routes in downtown Philadelphia, b.) The solution routes in Brookston, Indiana.

3.2.2 Contributions

In this chapter, we propose a new MIP model and a heuristic algorithm to solve a new problem, the multiple Traveling Salesman with Drones (mTSPD) problem. The main contributions of this work are the following:

1. We introduce a new variant of the Traveling Salesman Problem with Drones in which multiple trucks and drones are deployed to make deliveries. We call the problem "multiple Traveling Salesman with Drones" (mTSPD). The model is based on the Multiple Traveling Salesman Problem (Bektas, 2006) with the mathematical formulation adapted from the FSTSP model (Murray & Chu, 2015).
2. We develop a new heuristic called "Adaptive Insertion Heuristic" (ADI) to solve mTSPD. The heuristic builds truck-drone tours from the original constructed mTSP solution, which consists of only *m* truck tours. We propose three mTSP heuristics to be used for ADI and compare the performance among them.
3. We use the ADI heuristic to solve some generated small-scale (less than 10 nodes) test instances and compare the solution performance with mTSPD and other existing TSP/FSTSP models MIP formulation solved by CPLEX. The results give us an insight on the savings achieved by implementing the multiple trucks and multiple drones delivery system

4. We implement the ADI heuristic on several TSP/mTSP benchmark problems and compare its performance with the modified FSTSP heuristic, which is based on the heuristic developed to solve the FSTSP (Murray & Chu, 2015). The solutions of both heuristics are compared with the best known optimal TSP/mTSP solutions in each benchmark. The analysis of the results will show how much the logistics operator would benefit from implementing drones along with trucks to make the last-mile deliveries.

3.3 Mathematical Formulation

The mTSPD is defined on a directed graph $G = (V, E)$, where V is the set of n nodes representing customers with one depot and E is the set of arcs. Let $\tau^T_{i,j}$ be a truck travel time associated with E, $(i, j) \in E$ and $\tau^D_{i,j}$ be a drone travel time associated with E, $(i, j) \in E$. Differentiating the travel times for the truck and drone accounts for each vehicle's unique travel speed. The mTSPD is said to be symmetric if $\tau^T_{i,j} = \tau^T_{j,i}$ and $\tau^D_{i,j} = \tau^D_{j,i}$ and asymmetric otherwise. Let m be the number of trucks that depart and return to the depot. Next, denote the set of customer nodes by $C = \{1, 2, 3, 4, 5, 6, \ldots, n\}$. Although only one physical depot location exists, we assign it to two unique node numbers at $0(s)$, the starting depot, and $0(r)$, the ending depot. Set $C_0 = C \cup \{0(s)\}$ as the set of customer nodes including the starting depot and set $C_+ = C \cup \{0(r)\}$ as the set of customer nodes including the ending depot.

Following the formulation in Murray and Chu (2015), we define $F = \{(i, j, k)\}$ as all possible three-node sorties of the drone path. An element $(i, j, k) \in F$ if the following conditions hold: 1) The launch node i must not be the ending depot node (i.e., $i \in C_0$). 2) The delivery node j must be in the customer set and must not be the same as the launch point (i.e., $j \in C$ such that $i \neq j$). 3) The merging point k can be either a customer node or the ending depot and cannot equal i or j ($k \in C_+$ such that $k \neq i$ and $k \neq j$).

We define the following decision variables: Let $x_{i,j}$ equal to 1 if arc $(i, j) \in E$ is used on a truck path and 0 otherwise. This refers to the situation when the truck travels from node $i \in C_0$ to $j \in C_+$ where $i \neq j$. Let $y_{i,j,k}$ equal to 1 if arc (i, j) and $(j, k) \in E$ is used on the path and 0 otherwise. This refers to the situation when a drone is launched from node $i \in C_0$ to node $j \in C$ (visiting customer node) and merges with a truck or the ending depot at node $k \in C_+$ such that *(i, j, k)* $\in F$. We denote Tl_j as the truck arrival time at node $j \in C_+$ and $\mathrm{Dl_j}$ as the drone arrival time at node $j \in C_+$. Tl_j and Dl_j are the arrival times of the truck and drones at node *j* respectively. Lastly, the auxiliary decision variable u_i is used in the TSP sub tour elimination constraints (Desrochers & Laporte, 1991). All the mentioned notations can be summarized as follows.

<u>*Indices*</u>

i, j, k	Represent customers, and depot

<u>*Sets*</u>

C	Set of customers, $\{1,\ 2,\ 3,\ 4,\ 5,\ 6, \ldots, n\}$
C_0	Set of customer nodes including the starting depot, $C \cup \{0(s)\}$
C_+	Set of customer nodes including the ending depot, $C \cup \{0(r)\}$

<u>*Parameters*</u>

$\tau_{i,j}^{T}$	Truck travel time between nodes *i* and *j*
$\tau_{i,j}^{D}$	Drone travel time between nodes *i* and *j*
m	Number of trucks in the entire fleet
n	Number of total customers to be served

<u>*Variables*</u>

$x_{i,j}$	1 if a truck traverses arc (i, j) from customer *i* to customer *j*; otherwise, 0
$y_{i,j,k}$	1 if a drone traverses arc (i, j) and (j, k) from customer *i* to customer *j* and from customer *j* to customer *k*; otherwise, 0
Tl_j	Truck arrival time at node *j*
Dl_j	Drone arrival time at node *j*
u_i	$Auxiliary\ variables\ for\ subtour\ elimination$

The proposed MIP formulation of mTSPD is presented as follows.

Objective

$$\text{minimize } Dl_{0(r)} \tag{1}$$

Subject to

$$\sum_{\substack{i\in C_0\\ i\neq j}} x_{i,j} + \sum_{\substack{i\in C_0\\ i\neq j}} \sum_{\substack{k\in C_+\\ (i,j,k)\in F}} y_{i,j,k} = 1 \quad \forall j \in C \tag{2}$$

$$\sum_{j\in C_+} x_{0(s),j} = m \tag{3}$$

$$\sum_{i\in C_0} x_{i,0(r)} = m \tag{4}$$

$$u_i - u_j + n x_{i,j} + (n-2) x_{j,i} \leq n-1 \quad \forall i,j \in C, i \neq j \tag{5}$$

$$1 + (n-2) x_{i,0(s)} + \sum_{\substack{j\in C\\ i\neq j}} x_{j,i} \leq u_i \leq n - (n-2) x_{0(s),i} - \sum_{\substack{j\in C\\ i\neq j}} x_{i,j} \quad \forall i \in C \tag{6}$$

$$\sum_{\substack{i\in C_0\\ i\neq j}} x_{i,j} = \sum_{\substack{k\in C_+\\ k\neq j}} x_{j,k} \quad \forall j \in C \tag{7}$$

$$\sum_{\substack{j\in C\\ i\neq j}} \sum_{\substack{k\in C_+\\ (i,j,k)\in F}} y_{i,j,k} \leq 1 \quad \forall i \in C_0 \tag{8}$$

$$\sum_{\substack{i \in C_0 \\ i \neq k}} \sum_{\substack{j \in C \\ (i,j,k) \in F}} y_{i,j,k} \leq 1 \quad \forall k \in C_+ \tag{9}$$

$$2y_{i,j,k} \leq \sum_{\substack{h \in C_0 \\ h \neq i}} x_{h,i} + \sum_{\substack{l \in C \\ l \neq k}} x_{l,k} \quad \forall i,j \in C, \forall k \in C_+ \tag{10}$$

$$y_{0(s),j,k} \leq \sum_{\substack{h \in C_0 \\ h \neq k}} x_{h,k} \quad \forall j \in C, \forall k \in C_+ \tag{11}$$

$$\sum_{\substack{i \in C_0 \\ i \neq j}} \sum_{\substack{k \in C_+ \\ (i,j,k) \in F}} y_{i,j,k} \leq 1 - \sum_{\substack{a \in C \\ a \neq j}} \sum_{\substack{b \in C_+ \\ (j,a,b) \in F}} y_{j,a,b} \quad \forall j \in C \tag{12}$$

$$\sum_{\substack{i \in C_0 \\ i \neq j}} \sum_{\substack{k \in C_+ \\ (i,j,k) \in F}} y_{i,j,k} \leq 1 - \sum_{\substack{a \in C_0 \\ a \neq j}} \sum_{\substack{b \in C \\ (a,b,j) \in F}} y_{a,b,j} \quad \forall j \in C \tag{13}$$

$$Dl_{(i)} \geq Tl_{(i)} - M(1 - \sum_{\substack{j \in C \\ i \neq j}} \sum_{\substack{k \in C_+ \\ (i,j,k) \in F}} y_{i,j,k}) \quad \forall i \in C \tag{14}$$

$$Dl_{(i)} \leq Tl_{(i)} + M(1 - \sum_{\substack{j \in C \\ i \neq j}} \sum_{\substack{k \in C_+ \\ (i,j,k) \in F}} y_{i,j,k}) \quad \forall i \in C \tag{15}$$

$$Dl_{(k)} \geq Tl_{(k)} - M(1 - \sum_{\substack{j \in C_0 \\ j \neq k}} \sum_{\substack{j \in C \\ (i,j,k) \in F}} y_{i,j,k}) \quad \forall k \in C_+ \tag{16}$$

$$Dl_{(k)} \leq Tl_{(k)} + M(1 - \sum_{\substack{j \in C_0 \\ j \neq k}} \sum_{\substack{j \in C \\ (i,j,k) \in F}} y_{i,j,k}) \quad \forall k \in C_+ \tag{17}$$

$$Tl_{(k)} \geq Tl_{(h)} + \tau^{T}_{(h,k)} - M\left(1 - x_{h,k}\right) \quad \forall h \in C_0, \forall k \in C_+ \tag{18}$$

$$Dl_{(k)} \geq Dl_{(i)} + \tau^{D}_{(i,j)} + \tau^{D}_{(j,k)} - M\left(1 - y_{(i,j,k)}\right) \quad \forall i \in C_0, \forall j \in C, \forall k \in C_+ \tag{19}$$

$$Dl_{0(s)} = 0 \tag{20}$$

$$Tl_{0(s)} = 0 \tag{21}$$

$$x_{i,j} \in \{0,1\} \quad \forall i, \mathrm{j} \in C \cup C_0 \cup C_+ \tag{22}$$

$$y_{i,j,k} \in \{0,1\} \quad \forall i, \mathrm{j}, \mathrm{k} \in C \cup C_0 \cup C_+ \tag{23}$$

$$Dl_i \geq 0 \quad \forall i \in C \cup C_0 \cup C_+ \tag{24}$$

$$Tl_i \geq 0 \quad \forall i \in C \cup C_0 \cup C_+ \tag{25}$$

The objective function (1) minimizes the arrival time of drones and trucks at the depot. The objective function is equivalent to $min\left\{max\{Tl_{0(r)}, Dl_{0(r)}\}\right\}$ since both drone and truck have to wait for each other and the arrival time of trucks and drones at the depot will be adjusted to be the same by constraints (14) – (17). Constraints (2) ensure that each customer will receive the package either by a drone or truck. Constraint (3) and constraint (4) ensure that the trucks depart from and arrive to the depot. Constraints (5) and (6) are sets of the Desrochers and Laporte (DL) sub tour elimination constraint which ensures that there is no sub tour in all tours of the trucks (Desrochers & Laporte, 1991). Constraints (7) guarantee that whenever the truck arrives at a node, it must depart from the node as well. Constraints (8) represent that at most one drone can depart at each stop of the truck. Similarly, constraints (9) define that at most one drone can arrive to a truck if it visits a node. Without these constraints, there is no restriction on how many drones can be launched from and land on a certain node, which would violate assumption 5 in Section 3.2.1. Constraints (10) state that a truck must visit node *i* and node *k* if the drone is launched from node *i* and is retrieved at node *k*. Similarly, constraints (11) ensure that when the drone flies from the depot to node *j* and *k* respectively, a truck must correspondingly depart from the depot and eventually arrive at node *k*. Constraints (12) describe the cases that if the drone flies from node *i* to node *j* to node *k*, there are no other drones that make such a delivery simultaneously from node *j* to node *a* to node *b*. Constraints (13) enforce that if a drone departs from node *i* to visit node *j* and merges with the truck at node *k*, no other drones can arrive at the delivery node *j*. Constraints (12) and (13) will ensure the flow conservation for the delivery node. Constraints (14) and (15) state that the departure time of drone and truck must be the same. Also, once the drone and truck are in the same node, they must wait for each other before each of them can leave the node. For an example, let assume that both truck and drone are at node i. If in the next move, the drone travels from node i to j to k, then constraints (14) and (15) will be binding, resulting in the same departure time for both truck and drone. Similarly, constraints (16) and (17) ensure that the arrival time of both truck and drone will be the same when they merge at the same node. These sets of constraints are based on the assumption that if either the drone or truck arrives earlier than the other, the earlier one has to wait until the later one arrives (both constraints are binding, resulting in the same arrival time of both truck and drone). Constraints (18) keep track of the arrival time of the truck at every node. It adds the truck travel time to the previous customer node when the truck travels from one customer node to another customer node. Similarly, constraints (19) keep track of the arrival time of the drone at the node to which the drone returns after making a delivery. This constraint considers the drone's travel time from the previous departure node to the delivery node to the arrival node. Constraints (20) and (21) set the initial departure time of drones and trucks at the depot to be zero. Constraints (22–25) specify the types and ranges of the variables. Note that the M value must be large enough. Thus, we can use the minimum total travel time of a single truck to visit all customers in one single tour, i.e., solve a regular TSP.

We have tested the MIP formulation of the mTSPD using the CPLEX solver on small and medium-size (5-50 nodes) problems, which we generated based on the method in Section 3.4. Although we are able to obtain the optimal solutions within one hour on 10-node problems, the computational time becomes prohibitive (more than 8 hours) on larger-scale problems (more than 10 customer nodes). Since mTSPD is an extension of the well known TSP/mTSP which is an NP-hard problem, the problem itself is also an NP-hard as well. As a result, we have developed an Adaptive Insertion algorithm (ADI) to solve the mTSPD problem. We propose a modified version of the FSTSP heuristic, so-called "Adapted FSTSP heuristic" so that it can be used to solve the mTSPD problem. In the experiment section, we will compare the performance of our proposed heuristic with the FSTSP heuristic proposed by Murray and Chu (2015). The numerical results and related statistics will be provided in Section 3.5.

3.4 The Proposed Algorithm

3.4.1 Overview

The Adaptive Insertion algorithm, "ADI" was developed to efficiently solve the mTSPD. This heuristic is based on the greedy node(s)-insertion strategy (Gendreau, Hertz, & Laporte, 1992; Lu & Dessouky, 2006). Prior to developing the heuristic, we used the MIP model to solve the mTSPD, and we were able to solve and return the optimal solutions of the problems with up to 10 customer nodes within an hour. However, larger problem sizes could not be optimally solved even within 8 hours. Therefore, heuristic solution approaches are required to find optimal or near-optimal solutions once the problem size becomes larger than 10 nodes.

There are two phases in our proposed our heuristic. In the first phase, an initial mTSP solution which consists only truck tours is generated through a construction heuristic algorithm. In the second phase, we build a mTSPD solution from a mTSP solution using different types of removal and insertion operators. The mTSPD solution consists of the improved truck tours together with the drone tours constructed in the second phase. The algorithm is detailed in the following section.

3.4.2 Initial mTSP Solution

Since a mTSPD solution is based on the constructed tours of the mTSP solution in the first phase, different initial mTSP solutions can affect the solution quality of the mTSPD in the second phase. Therefore, we use three heuristics to generate the mTSP routes as follows:

<u>Genetic Algorithm (GA)</u>

The GA is based on the principles of natural selection and genetics to generate a better solution over the solution from the previous generation. It starts with a group of initial solutions. A fitness function is used to evaluate the performance of the solutions. Then, the two solutions called parent solutions are selected based on a certain probability to perform a crossover/mutation operation to produce two new solutions of the next generation. If the new solutions have better fitness values, they will replace the old solutions. The selection, crossover, and mutation operations are repeated for the rest of the population until the population size of the new generation is the same as the size of the previous generation. This completes one iteration (generation). The procedure continues until the number of certain generations is reached or the solution quality cannot be improved (Carter & Ragsdale, 2006; Li, Sun, Zhou, & Dai, 2013).

To implement GA in our problem, we have to modify the fitness function so that it can be used with the min-max mTSP. We use one order crossover and two mutation operators proposed by Sedighpour et al. (2011) in the GA. We also conduct some experiments with different sets of parameters, including selection probability, cross-over probability, mutation probability and the number of generations to ensure obtaining the high-quality solutions by the algorithm.

Control parameters

NumGen: Number of Generations

PopSize : PopulationSize (Number of chromosomes)

S: Current solution of the mTSP

f: The fitness value (objective value) of S

S^**: The best solution of the mTSP*

f^**: The best fitness value of* S^*

p_C*: Crossover probability*

p_m*: Mutation probability*

p_{select}*: Selection probability*

Proposed GA for mTSP

Load Input parameters

Create initial populations of the mTSP //Each chromosome represents a route solution//

Find the best *S*. Record the best solution as S^* with the fitness value of f^*

For the total *NumGen*

  For all *PopSize*

    Determine *f* for each solution route (chromosome)

  End For

  Keep the best 2 solutions without going to crossover

  Select parents to perform crossover with p_{select} (Roulette Wheel Selection)

  Perform Order crossover with p_C for all pairs of chromosomes

  Perform Two mutation operators with p_m for all chromosomes

Find the best *S* among the current populations.

If $f\,(Best\ S) \leq f^*$

$f^* = f\,(Best\ S)$

End If

End For

Combined K-means / Nearest Neighbor

This heuristic combines the K-means clustering algorithm with the well known nearest neighbor algorithm. K-means technique is quite popular among various clustering techniques because of its ability and efficiency among clustering data. Among the set of customer nodes (C), we partition them into *k* clusters. Please note that the notation *k* in this K-means algorithm is different from the notation *k* in Section 3.3 which is previously defined as an index for the customer node (i,j,k). Each cluster represents a tour of one truck. Initially, we randomly assign k cluster centroids into the map. Based on the distance from the centroid, each customer node is assigned to the nearest centroid. Each cluster centroid is updated based on the nodes assigned to the cluster. The process will be repeated until the centroids remain the same or no point changes clusters (Jain, 2010). Let X = $\{x_i\}, i = 1, \dots, n$ be the set of n customer nodes to be clustered into a set *K* cluster with the cluster mean, C = $\{c_k, k = 1, \dots, K\}$. The goal of k-means is to minimize the sum of the squared error over all k clusters,

$$J(C) = \sum_{k=1}^{K} \sum_{x_i \epsilon C_k} \| x_i - \mu_k \|^2 \qquad (26)$$

The K-means algorithm to generate K clusters is composed of the following steps:

Step 1: Randomly generate K points into the map represented by the customer nodes that are being clustered. These K points represent initial cluster centroids.

Step 2: Assign each node to the cluster that has the closest centroid.

Step 3: Once all nodes have been assigned to the clusters, recalculate the locations of the K centroids.

Step 4: Repeat steps 2 and 3 until the centroids no longer move. This procedure assigns the customer nodes into clusters which corresponds to minimize the equation (26).

After partitioning all customer nodes into k clusters, we then use the Nearest Neighbor (NN) to solve for the TSP tour in each cluster (Arora, Agarwal, & Tanwar, 2016). It is required that the starting node of each TSP tour must be a depot node and the ending node of each TSP tour must be a depot node. In NN, the truck starts at the depot, repeatedly visits the nearest customer node until all customers in the cluster are visited and returns back to the depot.

The Nearest Neighbor algorithm for the TSP solution is composed of the following steps:

Step 1: Set the depot as the current visited vertex. The rest of the customer nodes are listed in unvisited vertex set.

Step 2: Select the node V with the lowest travel cost and set it as a current vertex.

Step 3: Mark V as visited and update unvisited vertex set.

Step 4: If all the vertices in domain are visited, then terminate. The final visited vertex must be a depot.

Step 5: Repeat step 2- 4.

<u>Random Cluster / Tour</u>

In this heuristic, random *k* clusters are generated using a uniform distribution. Each customer node is uniquely assigned to one cluster. Each cluster must have at least one customer node and there is no restricted number of how many customer nodes can be assigned in a cluster. In each cluster, a truck starts at the depot, randomly visits all customer nodes in the cluster and returns back to the depot. The algorithm is composed of the following steps:

Step 1: Randomly generate K points into the map based on the number of trucks (m).

Step 2: Assign each customer node to the cluster. Each node can only be assigned to one cluster. Each cluster must be assigned at least one node.

Step 3: For each cluster, generate the random tour starting from the depot, visit each node exactly once and return to the original depot.

3.4.3 Construction of mTSPD Solution

Once obtaining the initial mTSP solution, we can construct the mTSPD solution by gradually replacing some of the truck customer nodes with drone customer nodes. The ADI algorithm gradually builds up a solution by switching the position of the nodes, reevaluating the objective value, and terminating once the termination condition is met. The heuristic seeks for the best way to remove the truck node from the tour and insert the node back to the tour as a drone customer node. We can perform the truck/drone tour construction by executing two main types of operators: the removal operator and the insertion operator. The removal operator takes one node out of the existing tours. This particular node will be sequentially added back to one of the tours via three types of insertion operators. The algorithm aims to maximize the *Saving* or the difference between "Time Decrease from removing node" (ΔT^{D}) and "Time Increase from adding node" (ΔT^{I}) in each iteration until the termination condition is met. We also define "*#NCS*" to keep track of the number of customers who are served by any drone.

As previously mentioned, the algorithm begins by generating the mTSP initial solution through the described heuristics in section 3.4.2. Denote the set *Tour = {1,2,3,…m}* and assign an index *select* to track all the tours from *select = 1,2,3,…m*. Next, we define the set of *CandList = {1,2,3,…,n}*. The members of the CandList are all the customers (C) with the size *n* on the map. The removal and insertion procedures are applied on each element of the *CandList*. Once all members in *CandList* complete these two consequential procedures, all the feasible solutions will be evaluated through the fitness function *max* $\{(\Delta T_j^{D} - \Delta T_j^{I})\}$; $\forall j \in$ *CandList* (Line 28 of Algorithm ADI). Then, node *j* with the best fitness value is selected and removed from the *CandList* set (Line 29 of Algorithm ADI). We also need to update the solution as well as other global variables including: *Saving, #NCS, all Truck Route (T_R), Drone Path (D_P),* and the

Algorithm: ADI

Initialize

Parameters: *m, n,* τ^T , τ^D

Generate the initial mTSP solution with *m* truck tours

Compute time (t_i^{select}) associated with each node *i* in tour *select*

Max #NCS = $\sum_i^m \lfloor \frac{(\#customers\ in\ tour\ i-2)}{2} \rfloor$

Create *T_R*= {*Tours,* t_i^{select}, *Lch_N, La_N*}

Set *Saving* = 1, *CandList = {All C}, Lch_N=[], La_N =[], D_P = [],* ΔT^S *= [], #D = 0*

While *Saving > 0*

    For *j* ∈ *CandList*

        Find a tour where node *j* belongs to; called *SelectTour*

        If *#C* in *SelectRoute < 2*

            ΔT^S *= -inf*

        Else

            Call *Node_Removal(T_R, D_P, j,* τ^T , τ^D *)*

            If *#NCS < Max #NCS*

                Call *DroneAddedSameRoute (T_R, D_P, j,* τ^T , τ^D *)*

                Call *DroneAddedDiffRoute (T_R, D_P, j,* τ^T , τ^D *)*

            Else

                $\Delta T1_j^I$ *= inf*

                $\Delta T2_j^I$ *= inf*

            End If

            Call *TruckInsertion(T_R, D_P, j,* τ^T , τ^D *)*

        End If

        $\Delta T_j^I = min(\Delta T1_j^I, \Delta T2_j^I, \Delta T3_j^I)$

        Select *T_R, D_P* associated with the node *j* in ΔT_j^I

        $\Delta T_j^S = \Delta T_j^D - \Delta T_j^I$

27. **End** For

28. Find *Saving* = $max_{j \in CandList}$ (ΔT_j^S)

29. Remove node *j* from *CandList*

30. Update: T_R *and* D_P associated with *Saving*

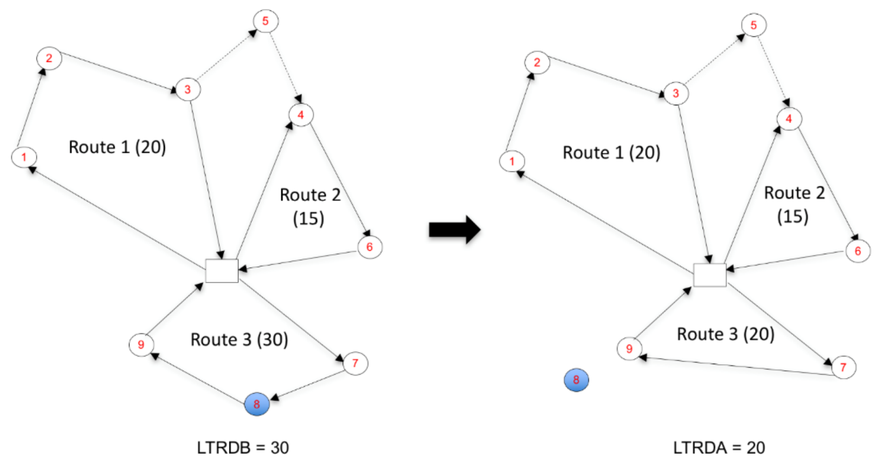
31. Update: *#NCS* and *CandList*

32. **End** While

arrival time associated with each customer node (t_j^{select}) (Line 30-31 of Algorithm ADI). For any truck path (*i, j, k),* we denote $t_k^{select*}$ as the arrival time at node *k* prior to removing node *j* from tour *select* and $t_k^{select**}$as the arrival time at node *k* after removing node *j* from tour *select.* We also keep track of the set of nodes where the drones are launched from using the notation *Lch_N* and the set of nodes where the drones land to using the notation *La_N*. The algorithm will repeat iteratively and terminate once $\Delta T_j^S = \Delta T_j^D - \Delta T_j^I \leq 0$ or *CandList* $\in \emptyset$. Please note that if *#NCS* ≥ *Max(#NCS)* (Line 15-16 of Algorithm ADI), we can only insert node *j* between nodes that are currently served by the truck. This constraint is used to prevent the algorithm from generating an excessive number of drone delivery paths with many infeasible solutions.

3.4.4 Removal Operator

Given the chosen node *j* from the main algorithm, the *Node_Removal* operator would determine how much objective value decreases when removing node *j* from the particular tour *select.* There are three main steps in the procedure. The first step is to remove node *j* from tour *select* and re-compute the arrival time to the depot at each truck. The next step is to find the arrival time to the depot of the last truck (latest arrival time at the depot) stored in the variable called Latest Time the Truck Returns to the Depot After removing node *j (LTRDA).* The last step is to find the difference between *LTRDA* and the Latest Time the Truck Return to the Depot Before removing node *j (LTRDB)*. The value difference is stored in variable ΔT_j^D**.** An example of the Removal operator is shown in Figure 3.3.



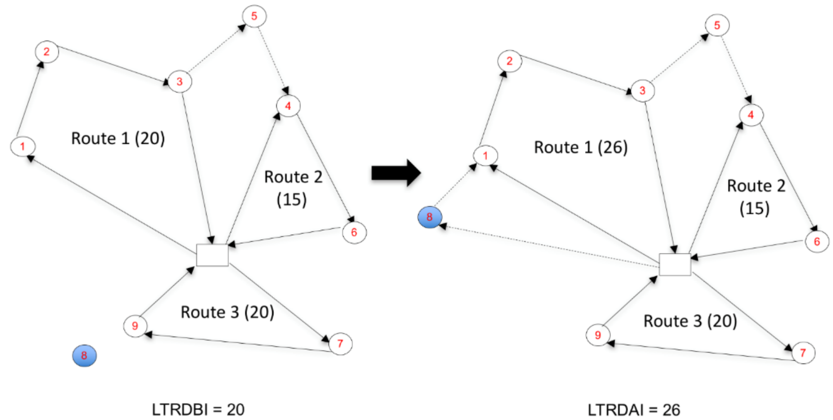
$$\Delta T_j^D = \text{LTRDB} - \text{LTRDA} = 10$$

Figure 3.3 Illustration of Removal operator

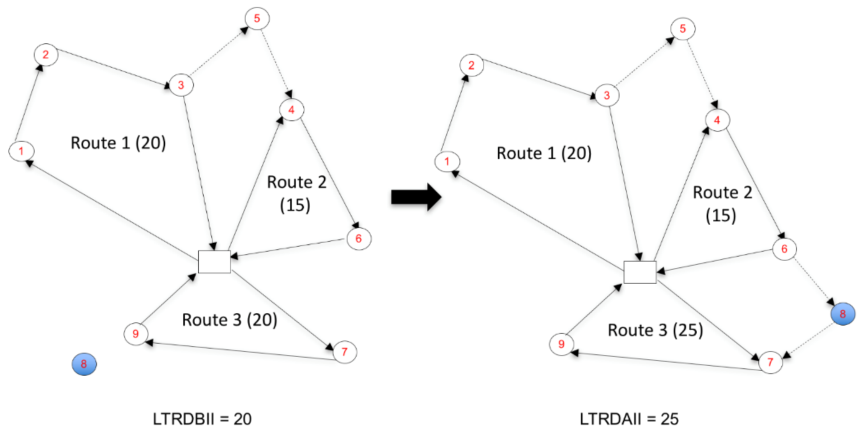
In the above example, the subfigure on the left represents the original routing solution for three trucks and the updated routing solution once applying the Removal operator. It takes 20 minutes, 15 minutes and 30 minutes for truck 1, truck 2 and truck 3 to complete the deliveries accordingly. We assign node 8 as the node to be removed from the existing solution. The Latest Time the Truck Return to the Depot Before (LTRDB) removing node 8 is 30 minutes which is the duration truck 3 completes its job. Once removing node 8 from the solution, the truck 3's delivery time is updated to 20 minutes. The new objective for the solution, the Latest Time the Truck Returns to the Depot After (LTRDA) removing node 8, is updated to 20 minutes. Hence, the difference between the objective value of the original and the updated solution is 10 minutes, which is stored in the variable ΔT_8^D.

3.4.5 Insertion Operator

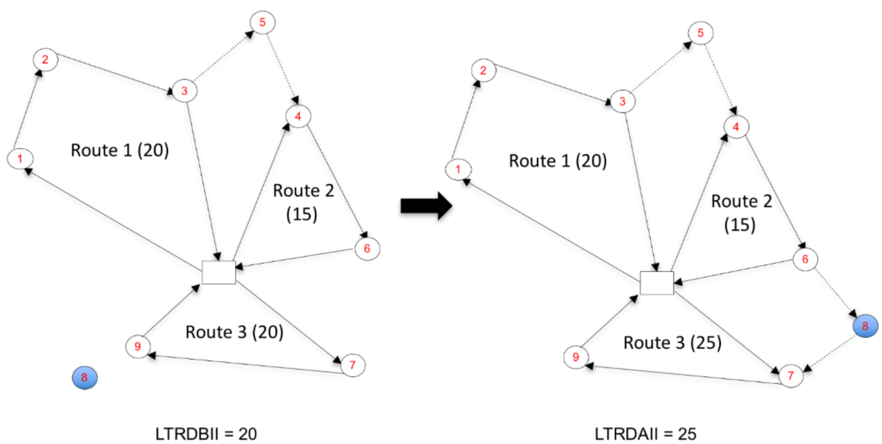
Once node *j* is removed from tour *select,* we need to find the location to insert node *j*. Three types of insertion algorithms are used for the search process: *DroneAddedSameRoute, DroneAdded-DiffRoute,* and *TruckInsertion*. In *DroneAddedSameRoute*, we assign node *j* as a drone delivery node and insert it into one of the *m* tours. The path *(i, j, k)* is constructed such that the drone departs from node *i*, serves the customer at node *j,* and merges with the truck at node *k* (the same truck that the drone is launched from). To explain briefly how this operator works, for each tour from all the truck tours, $Tour^1$, $Tour^2$, $Tour^3$,..., $Tour^m$, we select a pair of *(i,k)* nodes where node *i* must precede node k to insert node j into. After inserting node j between node i and node $k,$ we recalculate the arrival time when the truck returns on the selected tour, where node j is added. Finally, we update the latest arrival time at the depot for the whole route and calculate the difference between the Latest Time the Truck Return to the Depot After adding node *j (LTRDAI)* and the Latest Time the Truck Return to the Depot Before adding node *j (LTRDBI)*. This difference is stored in variable $\Delta T1_j^{I,p}$,where p is the index of all possible combinations of this type of insertion. The minimum of $\Delta T1_j^{I,p}$ is chosen among all possible combinations. Slightly different from *DroneAddedSameRoute*, the operator *DroneAddedDiffRoute* restricts that the drone path *(i, j, k)* must be constructed such that node *i* and node *k* are selected from different tours. Hence the drone would merge with a different truck at node *k*, not the one it was launched from, after visiting node *j*. The best difference in objective value before and after inserting node *j* into the route is stored in variable $\Delta T2_j^I$. Lastly, *TruckInsertion* deals with the case in which node *j* is assigned as a truck delivery node. Node *j* must be inserted between node *i* and node *k.* Both node *i* and node *k* must be located in the same tour, where node *i* must precede node *k* and the nodes must be adjacent to each other. These three operators provide different ways of searching and sorting methods. The basic idea of ADI is to look for the lower ΔT^I which tends to give the maximum possible *Saving*. We then select $min(\Delta T1_j^I, \Delta T2_j^I, \Delta T3_j^I)$ (Line 24 of Algorithm ADI) after completing the searches. Figure 3.4 illustrates all three types of Insertion operators.



a.) Illustration of DroneAddedSameRoute: $\Delta T1_j^I$ = LTRDAI – LTRDBI = 6



b.) Illustration of DroneAddedDiffRoute: $\Delta T2_j^I$ = LTRDAII – LTRDBII = 5



c.) Illustration of TruckInsertion: $\Delta T3_j^I$ = LTRDAIII – LTRDBIII = 7

Figure 3.4 Illustration of all types of Insertion operators

In the above example, all three subfigures represent the routing solution before and after applying three different types of operators. For the purpose of simplicity, we select node 8 as the node to be inserted into the original solution. In Figure 3.4a.), the original objective value is 20 minutes, which is the Latest Time the Truck Return to the Depot Before (LTRDBI) adding node 8 into the solution. The node 8 is inserted between the depot and node 7 as a drone delivery node of the truck 3's tour. The new delivery time of the truck 3 in Figure 3.4a.) becomes 26 minutes which is equivalent to the new objective value or the Latest Time the Truck Return to the Depot After adding node 8. Consequently, the difference between the objective value of the original and the updated solution is 6 minutes recorded in the variable $\Delta T1_8^I$. In Figure 3.4b.), the node 8 is inserted as a drone delivery node between the customer node 6 of truck 2 and the customer node 7 of truck 3, resulting in the new objective value of 25 minutes. The difference between the objective value of the original and the updated solution is 5 minutes recorded in the variable $\Delta T2_8^I$. Lastly, the node 8 is inserted as a truck delivery node between the customer node 6 and the depot of the truck 2's tour, resulting in the new objective value of 27 minutes (Figure 3.4c.). The difference between the objective value of the original and the updated solution is 7 minutes recorded in the variable $\Delta T3_8^I$. Among the three types of insertions, $\Delta T2_8^I$ returns the lowest cost of insertion and the following new solution from the second type of insertion is accepted.

3.5 Computational Examples and Results

Three experiments were conducted to evaluate the MIP formulation and the performance of the proposed algorithm. In the first experiment, we solved some generated random instances with a size of 25 and 50 customer nodes using Adaptive Insertion algorithm with three types of initial mTSP heuristics described in Section 3.4. The solutions of the ADI heuristic are compared with the solutions from CPLEX solver. The details of the instance generation will be explained in Section 3.5.1.

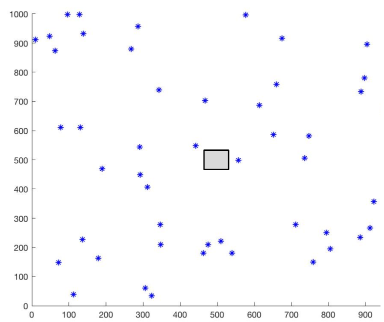
In the second experiment, we solved the small test instance (9 nodes) on different types of generated instances which were generated the same way as in the first experiment and compared the performance of our proposed algorithm with the optimal solution obtained by MIP. Using the same set of instances, we also use three different MIP formulations by CPLEX to solve them optimally and compare their solutions. These MIP models are: TSP (one truck), FSTSP (one truck and one drone) and mTSPD (multiple trucks and multiple drones).

In the last experiments, we demonstrate the performance of our algorithm on the well-known TSP/mTSP benchmark problems (Reinelt, 1991), which are relatively large and cannot be solved by CPLEX within a reasonable time. The solutions from the ADI are compared with the optimal solutions from the min-max TSP/mTSP problem, as well as the solutions from the Adapted FSTSP heuristic. Note that the Adapted FSTSP heuristic is modified from Murray and Chu's heuristic (Murray & Chu, 2015) to solve the mTSPD problem by implementing the FSTSP heuristic for each tour of the trucks and selecting the tour with the longest arrival time back to the depot.

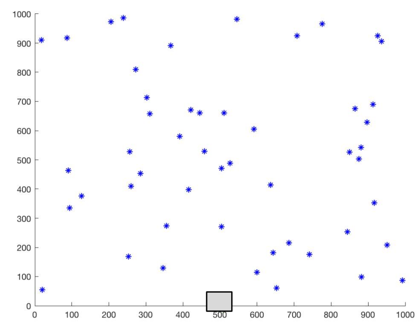
Regarding parameter setting, we set the truck travel time to be 1.5 time units longer than the drone travel time ($\tau_{j,i}^{T}$ = 1.5 $\tau_{i,j}^{D}$) since the drone speed is roughly about 1.5 times faster than the truck speed (Brar et al., 2015). We assume that both trucks and drones travel in Euclidean travel paths. All the algorithms were executed in Matlab on a computer with 2.7GHz Intel Core i5 with 8GB RAM running Windows 7 64-bit mode. All the Mixed-Integer Linear Programming models were solved using GAMS 23.51 with CPLEX solver.

3.5.1 Instance Generation

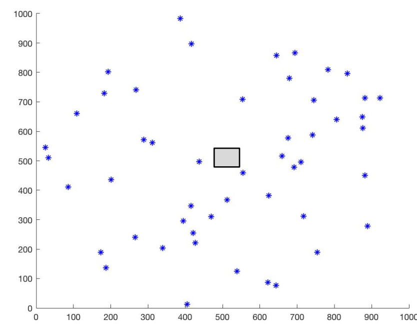
To evaluate the performance of the MIP model and ADI heuristic on the mTSPD, we generated 150 medium-size test instances (25, 50 nodes), and 35 small-size test instances (9 nodes). A set of customer nodes are created within the area of 1000*1000 $unit^2$. There are mainly five types of problems based on how the nodes are distributed on the map, as shown in Figure 3.5**.** In the type 1 problem, the depot is located at the center, at coordinate (500, 500) and surrounded by different customer nodes, which are uniformly distributed on the 1000 x 1000 square. The type 2 problem is similar to the Type 1 problem, except that the depot is now located at the bottom of the map at coordinate (500,0). In the type 3 problem, the customer nodes are uniformly generated on the circle area with a radius of 500. The depot is located at coordinate (500,500). On types I, II and III problems, we would like to see how the tour can be constructed when the customer nodes are located both close and far from the depot, and whether the location of the depot has any significant effect on the solution quality. In the type 4 problem, we assign the customer nodes to be far from the depot at coordinate (500,500) and distribute them in the ring pattern uniformly. To generate the customers in the ring area, we first create two circles: one with a radius of 500 and one with a radius of 300. The customer nodes are randomly generated in the area outside the small circle but within the big circle. Lastly, we created a cluster problem where the customer nodes are grouped as clusters. Each cluster has a circle shape with a radius of 125 and none of the clusters are overlapping. The last two problems are designed in the way that we can evaluate the solution of the routes when the customer nodes are located far away from the depot. In all problem types, we restrict that each node must be 20 units apart to prevent a situation in which two nodes locate too close or fall into the same location.



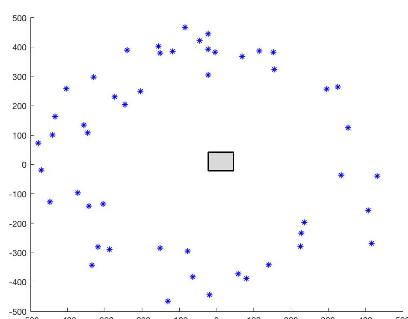
a.) Type 1: Center Depot surrounded by random nodes



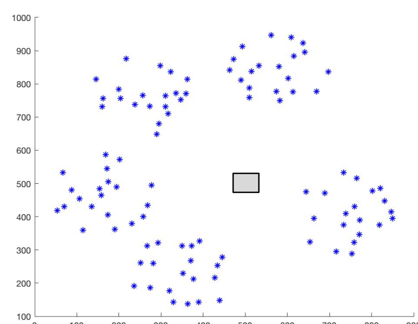
b.) Type 2: Bottom Depot with nodes on top



c.) Type 3: Center Depot surrounded by customers



d.) Type 4: Center Depot surrounded by customers in the ring pattern



e.) Type 5: Center Depot Cluster nodes

Figure 3.5 Different problem types based on the locations of depot and customer nodes.

3.5.2 Performance of ADI on Different mTSP Construction Heuristics in the mTSPD

Since the mTSPD problem is a newly-defined problem, there is no solution approach with which we can evaluate the performance of our heuristic framework. One of the ways to evaluate the performance of our heuristic is to compare its solution with the solutions obtained by solving the mTSPD using MIP which is an exact method. In this section, we evaluate the performance of ADI under three proposed mTSP construction heuristics: Genetic Algorithm, Combined K-means / Nearest Neighbor and Random Cluster / Tour, simply referred as GA-ADI, K-means ADI, and Random ADI accordingly. The experiment was conducted using different sets of instances generated from the previous section to evaluate the performance of the ADI and compare its solutions with the best solutions from the CPLEX. Let T1, T2, T3, T4 and T5 represent the instances of the problem type 1-5 accordingly. For each instance, we set different numbers of trucks from one to five. We ran this experiment on 25 customer nodes and 50 customer nodes (medium-size problem)0F[1]. Each combination of instance was run 20 times. With 5 instances, 5 number of trucks, 3 heuristics and 2 size problem set, we generated 150 tests. Since the formulated mTSPD is NP-hard, CPLEX is not able to obtain the optimal solutions within a reasonable amount of time for the medium-size problems. Therefore, we ran CPLEX for one hour (3600 seconds) for each instance and compare the results with the results from the proposed heuristic. We report the average objective value of each instance for all three heuristics and report the objective value found by CPLEX after running for an hour. The results are listed in Tables 3.1 and 3.2.

The results show that both GA-ADI and K-means ADI heuristics can obtain a much better solution results in significantly less time compared to CPLEX in both 25 and 50 nodes problem. On average, the GAP between these two heuristics and CPLEX is around 32%-34% for 25 nodes problem and around 73%-76% for 50 nodes problem. The results from the Random ADI are worse than CPLEX in terms of solution quality. It can be seen that both GA-ADI and K-means ADI outperform Random ADI which indicates that the use of good mTSP heuristics can improve the quality of ADI. Comparing between GA and K-means, the former slightly performs better than the latter by providing the lower (better) objective values in most of the instances. We also think that the solutions from GA can be improved with the right parameters in various settings. From this

[1] The generated instances can be accessed via https://github.com/pkitjach/Drone-Problem-Sets under "Medium Size Problem" folder

experiment, we decide to use GA to create the initial mTSP solution to construct the mTSPD solution in other experiments.

Table 3.1 Comparison the results between CPLEX and the ADI performance on different mTSP construction heuristics in 25 nodes mTSPD problem

Instance	# Trucks (m)	MIP		GA-ADI				K-means ADI				Random ADI			
		Obj	Time (sec)	Min Obj	Avg Obj	Avg GAP from MIP	Avg Time (sec)	Min Obj	Avg Obj	Avg GAP from MIP	Avg Time (sec)	Min Obj	Avg Obj	Avg GAP from MIP	Avg Time (sec)
T1	1	3868	3600	2788	2962.60	-23.41	43.11	2909	3047.55	-21.21	32.52	4693	5277.60	78.14	30.75
	2	2524	3600	1725	1816.05	-28.05	52.25	1646	1795.65	-28.86	40.73	2977	3410.55	87.80	48.90
	3	2135	3600	1350	1451.80	-32.00	65.46	1249	1453.35	-31.93	48.43	2106	2668.30	83.79	61.45
	4	1527	3600	1172	1243.55	-18.56	72.91	1273	1296.35	-15.10	56.17	1810	2262.85	81.97	67.43
	5	1338	3600	1133	1187.80	-11.23	77.93	1136	1199.85	-10.33	62.67	1490	1985.40	67.15	75.50
T2	1	5147	3600	2918	3076.10	-40.24	43.67	3170	3258.75	-36.69	29.35	4663	5594.55	81.87	29.37
	2	3916	3600	1887	2084.20	-46.78	53.45	2503	2580.30	-34.11	41.94	2859	3743.45	79.61	48.03
	3	2772	3600	1813	1970.80	-28.90	67.01	1834	2085.20	-24.78	47.76	2574	3101.80	57.39	56.36
	4	2573	3600	1757	1915.30	-25.56	76.78	1831	1957.95	-23.90	64.21	2295	2703.05	41.13	66.26
	5	2555	3600	1768	1842.95	-27.87	80.92	1869	1930.50	-24.44	66.68	2172	2481.70	34.66	70.82
T3	1	4130	3600	2349	2518.65	-39.02	40.26	2450	2561.80	-37.97	31.34	4329	4929.00	95.70	32.97
	2	2825	3600	1580	1655.50	-41.40	50.52	1695	1790.20	-36.63	41.07	2514	3059.55	84.81	51.31
	3	1741	3600	1282	1320.50	-24.15	55.14	1177	1270.5	-27.02	47.54	1935	2444.15	85.09	62.97
	4	1974	3600	1111	1155.55	-41.46	61.53	1072	1105.65	-43.99	55.90	1767	2005.40	73.55	71.85
	5	1768	3600	1045	1096.80	-37.96	64.13	1010	1037.65	-41.31	60.78	1376	1804.70	64.54	77.83
T4	1	6079	3600	2711	2825.85	-53.51	42.11	2757	2868.80	-52.81	31.92	3884	5592.90	97.92	32.95
	2	3014	3600	1721	1770.00	-41.27	52.12	1732	1765.70	-41.42	39.34	2786	3407.45	92.51	49.46
	3	2178	3600	1376	1441.85	-33.80	66.16	1345	1397.70	-35.83	46.81	2333	2933.80	103.47	58.98
	4	1966	3600	1210	1293.00	-34.23	71.84	1185	1215.95	-38.15	53.88	2082	2553.90	97.52	70.29
	5	1949	3600	1098	1198.95	-38.48	73.53	1167	1169.70	-39.98	54.39	1816	2224.30	85.52	77.77
T5	1	3053	3600	2021	2293.10	-24.89	45.09	2080	2329.00	-23.71	29.89	3407	4138.15	80.46	30.84
	2	2379	3600	1287	1317.60	-44.62	54.58	1287	1322.90	-44.39	39.10	2210	2689.75	104.14	46.72
	3	1828	3600	1087	1081.45	-40.84	60.50	1078	1104.35	-39.59	45.67	1792	2149.55	98.77	56.81
	4	1498	3600	957	982.30	-34.43	65.63	907	965.55	-35.54	49.04	1554	1945.10	98.01	68.02
	5	1436	3600	820	846.95	-41.02	71.14	1067	1126.45	-21.56	58.35	1467	1656.15	95.54	58.35
Mean			3600			-34.15	60.31			-32.45	47.02			82.04	56.08

Table 3.2 Comparison the results between CPLEX and the ADI performance on different mTSP construction heuristics in 50 nodes mTSPD problem

Instance	Trucks (m)	MIP		GA-ADI				K-means ADI				Random ADI			
		Obj	Time (sec)	Min Obj	Avg Obj	Avg GAP from MIP	Avg Time (sec)	Min Obj	Avg Obj	Avg GAP from MIP	Avg Time (sec)	Min Obj	Avg Obj	Avg GAP from MIP	Avg Time (sec)
T1	1	21766	3600	3712	4049.45	-81.40	92.99	3564	4017.10	-81.54	89.11	9957	10893.3	169.01	86.38
	2	12383	3600	2340	2500.95	-79.80	91.10	2472	2604.15	-78.97	92.21	6101	7177.30	186.98	110.75
	3	10721	3600	1801	2031.30	-81.05	103.69	1779	1971.05	-81.62	101.65	4503	5582.60	174.83	118.90
	4	6421	3600	1580	1778.90	-72.30	117.23	1532	1899.05	-70.42	107.18	3708	4641.90	160.94	128.63
	5	5296	3600	1410	1500.65	-71.66	128.08	1433	1495.95	-71.75	118.45	3221	4093.95	172.81	135.52
T2	1	25330	3600	3761	4297.75	-83.03	91.13	4175	4508.50	-82.20	88.45	10746	12091.2	181.34	87.22
	2	12942	3600	2537	2810.35	-78.29	98.80	2693	3041.35	-76.50	86.68	6091	7330.25	160.83	110.04
	3	9293	3600	2287	2428.00	-73.87	104.79	2336	2548.10	-72.58	99.46	5020	5819.15	139.67	121.06
	4	6622	3600	2134	2237.10	-66.22	106.60	2218	2349.40	-64.52	105.59	4194	5204.40	132.64	130.41
	5	5692	3600	2056	2168.50	-61.90	116.79	1970	2104.60	-63.03	111.39	3434	4686.30	116.11	137.83
T3	1	24094	3600	3526	3883.65	-83.88	91.17	3751	4027.25	-83.29	102.97	8301	10016.1	157.91	88.25
	2	11826	3600	2078	2310.60	-80.46	90.66	2259	2448.25	-79.30	103.43	4961	6128.70	165.24	113.16
	3	8342	3600	1595	1797.80	-78.45	98.90	1468	1794.18	-78.49	111.54	4127	4910.10	173.12	118.57
	4	6999	3600	1481	1571.40	-77.55	111.05	1406	1576.95	-77.47	111.58	3206	4098.70	160.83	130.23
	5	5469	3600	1250	1406.15	-74.29	124.85	1272	1447.85	-73.53	126.81	2935	3539.30	151.70	142.68
T4	1	23201	3600	3344	3541.25	-84.74	94.44	3343	3570.40	-84.61	87.41	9327	11526.7	225.50	86.85
	2	11381	3600	2110	2225.75	-80.44	92.99	2164	2315.15	-79.66	91.75	6135	7063.35	217.35	108.37
	3	8587	3600	1664	1876.40	-78.15	100.64	1791	1858.10	-78.36	96.87	4542	5355.85	185.43	115.76
	4	7306	3600	1537	1591.25	-78.22	111.35	1498	1545.50	-78.85	105.97	3583	4663.45	193.07	131.45
	5	6423	3600	1369	1403.40	-78.15	126.30	1335	1394.25	-78.29	116.59	3053	4046.60	188.34	139.74
T5	1	10945	3600	2768	2970.35	-72.86	94.87	3254	3378.75	-69.13	86.03	7635	9032.55	204.09	89.47
	2	6566	3600	1707	1882.65	-71.33	98.76	2964	3155.55	-51.94	109.67	4486	5763.65	206.15	108.32
	3	5281	3600	1442	1511.10	-71.39	105.24	1410	1687.95	-68.04	95.12	3498	4395.40	190.87	117.51
	4	4166	3600	1239	1336.70	-67.91	110.12	1409	1456.40	-65.04	116.29	2975	4106.65	207.22	123.59
	5	3752	3600	1045	1215.30	-67.61	125.02	1394	1447.45	-61.42	117.22	2606	3478.35	186.21	145.15
Mean			3600			-75.80	105.1			-74.02	103.18			176.33	117.03

3.5.3 Comparison of Different MIP Models and ADI Heuristic on Small-Size Problems

In this experiment, we want to evaluate the performance of the ADI heuristic when comparing with the exact solution from the MIP solver, which is only applicable to small-scale problem instances. We test the performance of the mTSP-ADI algorithm on different types of generated problems to examine the quality of the solutions. Beside comparing the solutions of the ADI heuristic with the optimal mTSPD solutions obtained from the MIP solver, we also compare the mTSPD solutions with solutions from the TSP and FSTSP models to evaluate the potential gain of implementing mTSPD model over the related last-mile delivery models. We solved all MIP formulations using CPLEX to obtain the optimal solution.

A collection of 35 test instances[2] was generated (7 instances for each problem type, i.e. T1A, T1B,…,T1G for Type1 problem). Each instance contains 8 customer nodes and one depot node distributed in the area based on the problem type described in the previous Section 3.5.1. For mTSPD, we only use one truck in the operation but allow multiple drones in the setting. For FSTSP, only one truck and one drone can be used in the operation. Finally, only a truck can be used in TSP model. We ran ADI heuristic 20 times for each instance and report the Best Objective, Average Objective and Average Time (seconds) of these 20 runs. We also report the Best GAP (%) and Average GAP (%) which is the difference between the Best ADI objective and the Optimal Objective and the difference between the Average ADI objective and the Optimal Objective, both in term of percentage accordingly. We are able to find the optimal objective values for TSP-MIP and mTSPD-MIP after a short period of time and report the objective value of the FSTSP-MIP solution after running the solver for 3,600 seconds.

The column Best GAP (%) reported in Table 3.4 shows that ADI can find all optimal solutions (except in T5D-T5G) while consuming significantly less computational time than the mTSPD MIP formulation. The Average GAP is about less than 1% on most instances indicating that the algorithm performs quite well in all small-size problems. The results of the comparison among different MIP models show that mTSPD outperforms the TSP and FSTSP with a gap value of -39.82% and -18.80% on average accordingly. These values indicate that on average the total time to complete the delivery by using one truck and multiple drones is 39% faster than the total delivery time using the truck alone and around 18.80% faster than the total delivery time using one truck and one drone. The gap is calculated by GAP = $100\left(\frac{\text{Mean mTSPD Obj}-\text{Mean TSP/FSTSP Obj}}{\text{Mean TSP/FSTSP Obj}}\right)$ which is shown in the last two rows of Table 3.4.

3.5.4 Robustness of Proposed Algorithm

In this section, we want to verify the robustness of the proposed algorithm in various settings. Using the same problem instances presented in Section 3.5.3, we ran the ADI on three initial mTSP construction heuristics and measure the optimality gap among the three of them. We report the GAP for each problem type (T1-T5) on each of the three mTSP heuristics: GA, K-mean/NN and

[2] The generated instances can be accessed via https://github.com/pkitjach/Drone-Problem-Sets under “Small Size Problem” folder

Random. Additionally, we ran the algorithm 20 times on each problem, and record the standard deviation (STD) of each problem. The results are demonstrated in Figure 3.6 and Table 3.3 below:

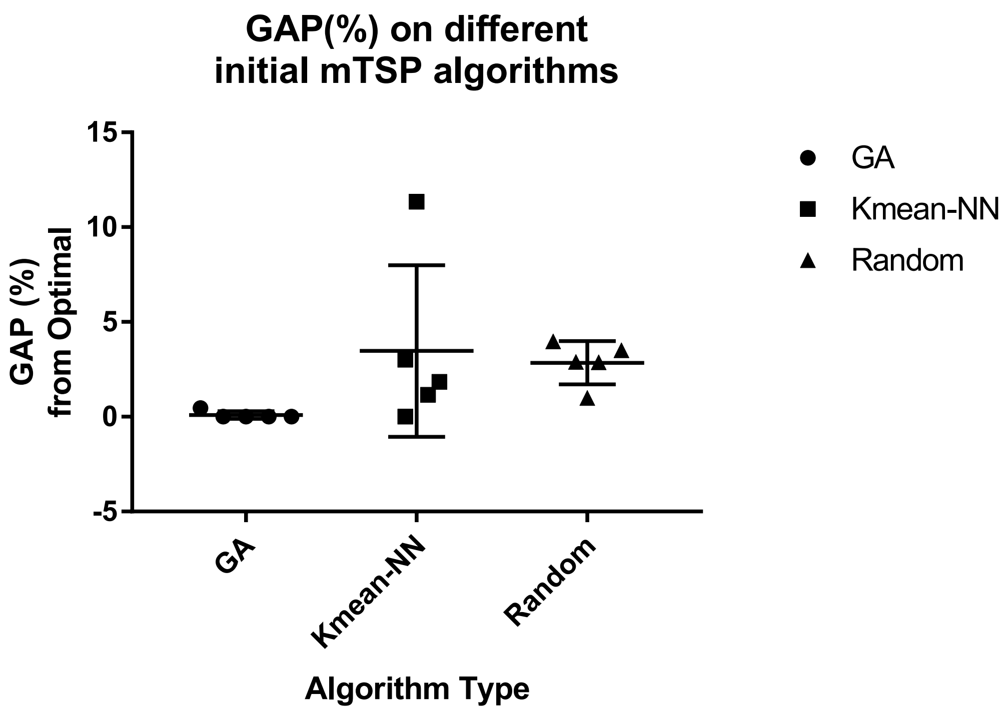


Figure 3.6 Optimal GAP (%) of ADI on different mTSP construction heuristics

Table 3.3 Relative standard deviation of each problem type among

Problem Type	Relative STD (%)		
	GA	Kmean-NN	Random
T1	0.51	2.37	7.58
T2	0.37	0.59	5.69
T3	1.04	1.74	8.45
T4	1.79	3.32	6.43
T5	0.53	0.91	3.36

The chart in Figure 3.6 demonstrates that the percentage gap for ADI-GA is zero for all types of problems which indicates that the algorithm is able to find the optimal solution in the small-size problems. On the contrary, ADI with Kmean-NN performs quite poorly with the optimal gap ranging from 0% to 13% depending on the problem type. It is possible that the search starting with mTSP solution from Kmean-NN often gets stuck with the local optimal solution and can only reach the global optimal solution in only certain types of problems. Lastly, the ADI is actually performing quite well when starting with random mTSP solutions in the small-size problem partly because of the small search space based on the problem size. From the relative standard deviation table, the standard deviation (STD) is relatively small (<2%) compared to the average value for all problem types. These STD values are higher for ADI with Kmean-NN and Random heuristics accordingly. Therefore, this experiment shows that the ADI-GA algorithm is robust against different problem types from the consistent zero optimality gap and small standard deviation.

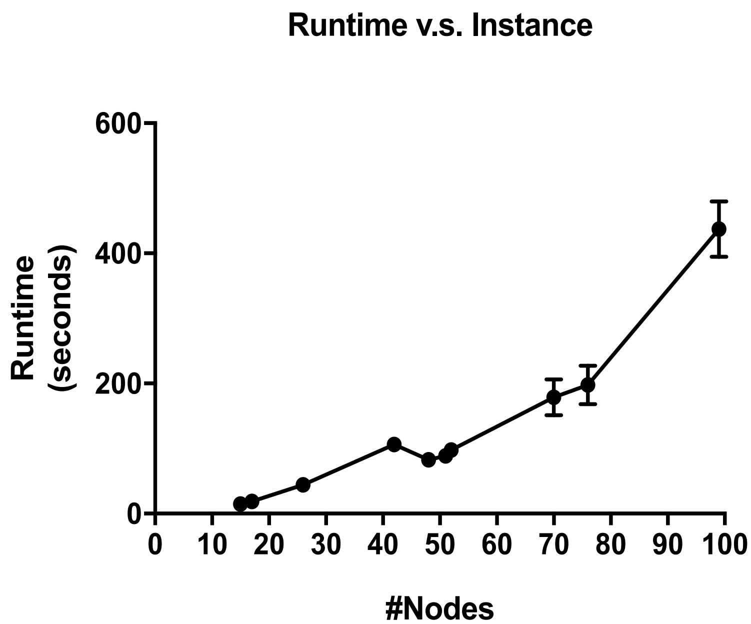
Table 3.4 Comparison of the results obtained from ADI heuristic and different MIP models on small-size problems

Instance	TSP-MIP	FSTSP-MIP		mTSPD						
				MIP		ADI				
	Opt Objective	Objective	Time (sec)	Objective	Time (sec)	Best Objective	Best GAP (%)	Average	Average GAP (%)	Average Time (Sec)
T1A	2594	1702	3600	1442	111.30	1442	0	1458.35	1.13	4.01
T1B	2947	1930	3600	1422	60.40	1422	0	1423.65	0.12	4.25
T1C	2488	1764	3600	1350	62.42	1350	0	1350.00	0.00	4.61
T1D	2800	1929	3600	1469	63.63	1469	0	1469.00	0.00	7.84
T1E	2655	1673	3600	1193	112.22	1193	0	1198.30	0.44	5.63
T1F	1773	1325	3600	1105	83.45	1105	0	1105.00	0.00	5.44
T1G	2255	1541	3600	1186	67.49	1186	0	1186.00	0.00	7.53
T2A	2578	2050	3600	1932	85.33	1932	0	1937.90	0.31	5.26
T2B	3036	2130	3600	1662	92.45	1662	0	1662.00	0.00	5.38
T2C	2498	2143	3600	1725	87.40	1725	0	1733.95	0.52	5.06
T2D	2702	2315	3600	1721	76.58	1721	0	1721.00	0.00	5.76
T2E	2703	2206	3600	1509	67.54	1509	0	1509.00	0.00	4.83
T2F	2989	2205	3600	1880	73.44	1880	0	1882.85	0.15	5.82
T2G	2633	2221	3600	1925	74.63	1925	0	1925.00	0.00	8.08
T3A	2849	1668	3600	1181	86.20	1181	0	1181.00	0.00	5.07
T3B	2556	2027	3600	1481	73.68	1481	0	1498.85	1.21	5.57
T3C	1832	989	3600	647	80.30	647	0	647.35	0.05	5.44
T3D	2383	1650	3600	1261	71.69	1261	0	1272.75	0.93	4.46
T3E	2324	1589	3600	1148	118.45	1148	0	1148.00	0.00	5.29
T3F	2615	1642	3600	1426	73.44	1426	0	1426.00	0.00	5.74
T3G	2576	1710	3600	1509	76.01	1509	0	1512.90	0.26	6.41
T4A	2479	1880	3600	1613	77.60	1613	0	1636.45	1.45	3.97
T4B	2625	2054	3600	1563	71.27	1563	0	1575.45	0.80	4.32
T4C	2005	1543	3600	1481	79.88	1481	0	1498.85	1.21	5.57
T4D	2428	1726	3600	1548	78.27	1548	0	1552.50	0.29	4.46
T4E	2348	1848	3600	1589	97.25	1589	0	1613.60	1.55	6.27
T4F	2699	2123	3600	1426	74.13	1426	0	1426.00	0.00	5.74
T4G	1942	1667	3600	1358	74.47	1358	0	1360.25	0.17	7.02
T5A	1405	1136	3600	1066	93.22	1066	0	1066.00	0.00	5.70
T5B	1322	1059	3600	944	135.60	944	0	944.00	0.00	4.54
T5C	1322	1015	3600	878	131.76	878	0	878.00	0.00	4.67
T5D	1348	1190	3600	1133	123.22	1138	0.44	1138.50	0.49	4.68
T5E	1351	1056	3600	1008	77.90	1014	0.60	1018.55	1.05	4.31
T5F	1579	1315	3600	1270	95.32	1283	1.02	1287.80	1.40	4.38
T5G	1518	1383	3600	1188	140.21	1202	1.18	1215.80	2.34	3.75
Mean	2290	1697		1378		1379		1385		
GAP(%) from TSP	-	-25.89		-39.82						
GAP(%) from FSTSP	34.94	-		-18.80						

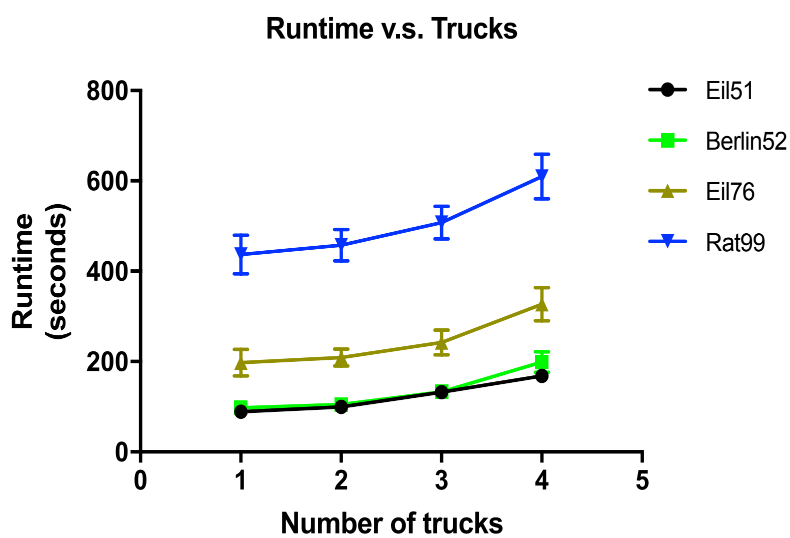
3.5.5 Performance of the Larger Instances on Benchmark Problem

In this experiment, we evaluate the performance of the ADI heuristic on the well-known TSP/mTSP benchmark problems from TSPLIB (Reinelt, 1991) with the min-max objective function. Each instance was solved with mTSPD-ADI with the change of the number of trucks {1, 2, 3 and 5}. We ran 20 replications for each instance with different numbers of trucks. We compare the average objective value obtained from the proposed ADI algorithm with the optimal objective of min-max TSP/mTSP for each instance. Since the TSP/mTSP does not allow the drone in the operation, the purpose of this experiment is to show how much operational time we could save by implementing multiple drones along with multiple trucks delivery strategy. We also tested the same set of instances with the Adjusted FSTSP heuristic and compared the performance of the algorithm with ADI.

The results are shown in Table 3.5. Overall, both the mTSP-ADI and Adjusted FSTSP heuristics return lower objectives value than the mTSP optimal values, as indicated by the negative values of GAP (%) from Optimal mTSP column. On average, the ADI heuristic returns the average gap lower than the min-max TSP/mTSP optimal solution by 20.95%. The gap is calculated by $GAP = 100(\frac{Avg\ Obj - Optimal\ Obj\ of\ TSP/mTSP}{Optimal\ Obj\ of\ TSP/mTSP})$. The negative sign on GAP (%) indicates a better objective value from the optimal objective of TSP/mTSP on both ADI and Adapted FSTSP Heuristic. Similarly, the Adjusted FSTSP heuristic returns the solutions better than the min-max TSP/mTSP optimal solutions by 8.14%. This finding demonstrates that assigning drones along with trucks to make deliveries is more effective than purely using trucks to make deliveries. Although the heuristic might not return the optimal solution, the high average gap still shows that it is more beneficial to implement multiple drones in the operation. It can be seen that the effect of drones become less significant once the number of trucks increase indicated by the GAP closer to zero. In addition, the ADI heuristic obtains a lower gap value than the Adjusted FSTSP heuristic's gap (about 13.84% on average), which can be noticed in the last column of Table 3.5. This value indicates that allowing multiple drones in delivery operations can significantly accelerate deliveries. In terms of computational time, Figure 3.7a.) shows that the runtime of the ADI heuristic rises quickly as the number of nodes increases. Similarly, Figure 3.7b.) demonstrates the increase in the runtime once the number of trucks increase on different instances



a.)



b.)

Figure 3.7 Run time based on a.) the Runtime v.s. Number of nodes and, b.) The Runtime vs. Trucks plot.

Table 3.5 Results of the ADI and Adapted FSTSP heuristics solutions on the benchmark problems

Instance	# Trucks (m)	Optimal Objective for TSP/mTSP	ADI				Adapted FSTSP Heuristic				GAP (%) between heuristics
			Avg Obj	GAP (%) from Optimal mTSP	Avg Customers served by Drones	Avg Time (s)	Avg Obj	GAP (%) from Optimal mTSP	Avg Customers served by Drones	Avg Time (s)	
p01	1.00	284.38	203.92	-28.29	5.00	15.03	246.17	-13.43	2.75	7.42	17.16
gr17	1.00	2085.00	1494.63	-28.31	5.45	18.85	1833.85	-12.05	1.80	7.71	18.50
fri26	1.00	937.00	732.55	-21.82	8.65	44.35	857.15	-8.52	2.55	9.05	14.54
dantzig42	1.00	699.00	475.07	-32.04	14.00	80.38	632.61	-9.50	3.70	13.72	24.90
att48	1.00	33524.00	22485.09	-32.93	16.00	82.87	30291.50	-9.64	3.55	15.92	25.77
st70	1.00	675.00	475.01	-29.63	23.00	178.57	633.21	-6.19	4.10	43.69	24.98
eil51	1.00	426.00	292.35	-31.37	17.00	88.90	380.6	-10.66	3.80	14.27	23.19
	2.00	222.73	173.44	-22.13	16.55	99.93	198.80	-10.74	5.10	15.79	12.76
	3.00	159.57	137.37	-13.91	15.15	132.45	140.10	-12.20	6.60	16.48	1.95
	5.00	123.96	108.64	-12.36	14.25	168.08	109.90	-11.34	8.10	16.57	1.15
berlin52	1.00	7542.00	4818.15	-36.12	17.00	97.82	6882.37	-8.75	3.65	24.23	29.99
	2.00	4110.21	3003.30	-26.93	16.40	104.96	3612.04	-12.12	4.75	20.71	16.85
	3.00	3244.37	2508.66	-22.68	15.30	133.25	2704.92	-16.63	6.55	21.60	7.26
	5.00	2440.92	2069.87	-15.20	13.90	198.67	1975.74	-19.06	8.05	21.11	-4.76
eil76	1.00	538.00	374.23	-30.44	25.00	197.97	513.59	-4.54	3.80	40.34	27.13
	2.00	280.85	230.83	-17.81	24.55	209.10	278.31	-0.90	5.30	48.27	17.06
	3.00	197.34	185.70	-5.90	23.40	242.00	196.90	-0.22	7.00	57.77	5.69
	5.00	150.30	148.05	-1.50	21.10	326.84	144.48	-3.87	10.40	75.42	-2.47
rat99	1.00	1211.00	946.36	-21.85	33.00	437.30	1225.08	1.16	4.25	94.25	22.75
	2.00	728.71	643.71	-11.66	31.70	457.58	736.68	1.09	5.80	157.56	12.62
	3.00	597.55	546.93	-8.47	30.00	507.87	581.67	-2.66	8.05	96.68	5.97
	5.00	531.87	480.72	-9.62	27.35	609.80	487.56	-8.33	11.15	97.29	1.40
	Mean			-20.95				-8.14			13.84

71

3.6 Conclusion

In this chapter, we propose a new routing model, the multiple Traveling Salesman Problem with Drones (mTSPD), which implements both trucks and drones in the last-mile delivery. The model is a variation of the classic TSP problem and the extension of the previous FSTSP model. We generalize the FSTSP to the mTSPD in which we allow multiple drones and multiple trucks to perform deliveries. The MIP formulation is mathematically constructed to model the mTSPD based on the delivery scenario described in Section 3.2. To solve the large-scale instances, we develop an Adaptive Insertion algorithm (ADI) which builds up the tours from the initial mTSP solution. The algorithm relies on removing an existing node from the route and adds it back to the route using three types of insertions. Three types of experiments are conducted to test the solution quality of the MIP model as well as the performance of the proposed algorithm. We conduct the first experiment by comparing the solutions of the ADI generated from different mTSP heuristics in the medium-size generated problems. The result in Section 3.5.2 has shown that the GA and K-means return the much better objective value than the solutions from CPLEX in terms of both solution quality and running time. This experiment leads us to conclude that a good mTSP heuristic leads to a better mTSPD solution. In the second experiment, we evaluate the performance of our heuristic by comparing its solution with the solutions obtained by solving the mTSPD via MIP solver in the small-size generated problems. The result in Section 3.5.3 shows that the mTSPD-ADI algorithm is able to return the solutions with the best gap equal to zero in all of the problem types. By comparing the results of the mTSPD with other existing last-mile delivery models (TSP/FSTSP), we can see that the delivery time can be significantly improved by using multiple drones in the route planning. Lastly, we conduct a new experiment in which we use our proposed heuristic to solve some of the benchmark min-max TSP/mTSP instances from TSPLIB. The obtained solutions are compared with the optimal solution of these benchmark problems. We find that the algorithm returns solutions with a lower average objective when compared with the optimal objective of the truck-only operation on the benchmark instances. In addition, the mTSPD-ADI returns better solutions than the ones solved by an Adapted FSTSP heuristic. The experimental results show that using multiple drones and trucks provides shorter delivery completion time than simply using trucks alone, multiple trucks, and a single truck-drone in the operation.

CHAPTER 4. TWO ECHELON VEHICLE ROUTING PROBLEM WITH DRONES

4.1 Problem Description

The Two Echelon Vehicle Routing Problem with Drones (2EVRPD) is a variant of a Vehicle Routing Problem with the implementation of drones in the operations. The problem aims to find the optimal set of routes for a fleet of vehicles to deliver packages to a given set of customers. We name this problem 2EVRPD since the problem utilizes the advantages of small (drone) and large (truck) vehicles in a delivery system. In the classical 2EVRP, the delivery is divided into two levels: where the trucks operate on the first level between a central depot and selected intermediate distribution facilities, called satellites and where the secondary vehicles at satellites serve all customers at the second level (Cuda et al., 2015). In the first echelon, a fleet of trucks must leave a depot, visit customers along the route and return to the same depot. The route constructed by each truck is referred to as *a truck route.* In 2EVRPD, trucks behave like intermediate depots where they can launch and retrieve drones at one of the delivery locations. Each truck is equipped with drones, which belong to a certain truck specifically. The drone can be launched from the truck and make multiple deliveries before returning to the truck. Once launching the drone, the truck can simultaneously make other deliveries along the route. Unlike the typical truck-drone delivery (FSTSP/TSP-D) where a drone can only make a single delivery and has to return to trucks immediately, a drone can make multiple deliveries per trip before returning to the truck which creates a *drone sub route* in each drone trip. All drone sub-routes are categorized as a second echelon delivery level. Please note that the termed "*multiple drops*", allows drones to make multiple deliveries at different customers before merging with trucks. To the best of our knowledge, we only see this feature available in drone-only routing problems or truck-only delivery but not in the truck-drone synchronized routing problem. This makes the problem much more complex and challenging as it integrates two VRPs into one truck-drone delivery routing problem with the consideration of the capacities of both trucks and drones as well as the order of launching and landing operations. Successful practical application of this feature in the industry could potentially bring about cost efficiency and reduce the total delivery time of the last-mile delivery.

The 2EVRPD aims at finding a set of both drone and truck routes such that the demand of all customers is satisfied and the constraints such as the truck's capacity, drones load capacity and drones battery capacity are not violated, while the total distribution time is minimized. The total distribution time is given by the total truck delivery time for all routes. In the 2EVRPD, both drones and trucks are capacitated, homogeneous within the same echelon. The fleets of vehicles are assumed to be unlimited; however, each truck has limited space to carry only a specific number of drones in it. A drone has its own capacity and so does a truck. A drone has a limited amount of battery capacity which determines how long it can travel before having to return to trucks for a battery swap or a recharge. Each customer has to be served by exactly once either by a truck or a drone (i.e., customer demand cannot be split). In addition, the time of both truck and drone at the customer locations must be adjusted to be the same. And finally, the drone must be back to the truck where it is launched from and it is important to keep track of when the drones are available for the launch. There is no limit on how many times drones can be launched but one must complete its sub route before initiating a new sub route. Figure 4.1 shows an example of a feasible solution for a 2EVRPD. The blue circle represents the depots while the red circles are the customers. The routes belonging to the truck's delivery are represented as solid lines, whereas the routes belonging to the drone's delivery are depicted as dotted lines.

4.2 Assumptions and Contributions

4.2.1 Assumptions

The following assumptions were considered for this problem:

- Customer coordinates or locations and customer demands are predetermined in advance;
- Drones can only merge with a truck at a customer node and are not allowed to merge with a truck in any intermediate location;
- Trucks and drones must wait for each other whenever one arrives at the customer node before the other;
- Multiple drones can fly simultaneously; however, only one drone can be launched or retrieved at each particular node at the same time;

- The set-up and recovery times when the drone is launched or retrieved at a particular node can be negligible since their values are small compared to the truck and drone travel time;
- Each truck is allocated to one depot and vehicle capacity limits the number of customers to be served by the depot. Similarly, each drone is specifically assigned to a particular truck and can not be merged with different trucks;
- The truck and drone do not necessarily follow the same distance metric.
- All trucks/drones are homogenous and travel at the same speed; and
- FAA regulations about the visual line-of-sight (VLOS) can be relaxed by allowing a first-person-view piloting.

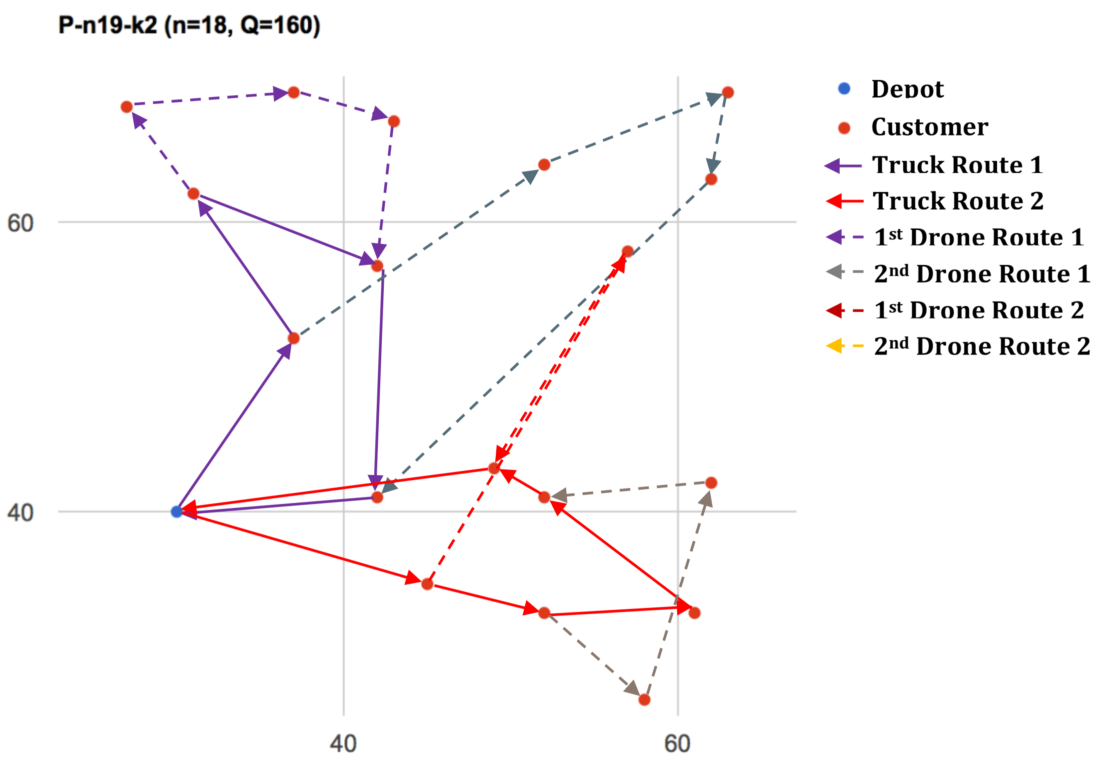


Figure 4.1 Illustration of the Two Echelon Vehicle Routing Problems with Drones (2EVRPD)

Finding good combined decisions for routing of both trucks and drones while satisfying all the constraints is significantly more difficult than the classical capacitated vehicle routing problem (CVRP) and other existing routing problems which involve drones. The problem 2EVRPD is considered as a variant of CVRP with a secondary level of delivery and is considered as the generalization of the FSTSP and our previous work mTSPD with capacity constraints and multiple drops. The 2EVRPD is a generalization of the classical VRP/TSP and is thus by nature an NP-hard. Mixed-Integer Programming formulation was developed to obtain an optimal solution that does work for the small-size problems. Because of the NP-hardness of the 2EVRPD, a heuristic approach is implemented to find solutions quickly for the larger-size problems. This study proposes two heuristic approaches: 1.) one based on a simple greedy approach called Drone Truck Route Construction (DTRC) and 2.) another one based on Large Neighborhood Search (LNS). Both were tested with the medium and large-size benchmark instances with the results provided in Section 4.5.

4.2.2 Main Contributions

In this chapter, we propose a new MIP model and heuristic algorithms to solve a new problem, the Two Echelon Vehicle Routing Problems with Drones (2EVRPD). The main contributions of this paper are the following:

1. We introduce a Mixed-Integer Program (MIP) formulation for the 2EVRPD. The model might be solved by any standard MILP solver, e.g., GAM and IBM CPLEX. They can handle small-size problems.
2. We propose two heuristics to solve 2EVRPD: 1.) Drone Route Construction (DTRC) and 2.) metaheuristic based on Large Neighborhood Search. In particular, DTRC generates an initial 2EVRPD solution from the VRP solution. The LNS will repeatedly search for a better solution using three destroy operators and three repair operators.
3. We conduct the numerical experiments and sensitivity analysis on different types of problems using both MIP and heuristics approaches. The small-size problems can be solved directly by the MIP solver and the large-size problems can be solved by DTRC and LNS. The results are compared with the classical VRP optimal solutions and other TSP/VRP with drones routing models on the CVRP benchmark problems.

Through our numerical study, the results show that the application of drones combined with trucks can significantly reduce the total travel time which makes them a valuable prospect for future last-mile delivery applications.

4.3 Mathematical Formulation

The 2EVRPD is defined on a directed graph *G = (V, E)*, where *V* is the set of *n* nodes representing customers with one depot and *E* is the set of arcs. Unlike the typical 2EVRP where the set of edges E is divided into two subsets, representing the first and the second echelon respectively, both echelon levels use only one set of edges E. Let $\tau_{i,j}^{T}$ be a truck travel time associated with *E,* $(i,j) \in E$ and $\tau_{i,j}^{D}$ be a drone travel time associated with *E,* $(i,j) \in E$. Differentiating the travel times for the truck and drone accounts for each vehicle's unique travel speed. The 2EVRPD is said to be symmetric if $\tau_{i,j}^{T} = \tau_{j,i}^{T}$ and $\tau_{i,j}^{D} = \tau_{j,i}^{D}$ and asymmetric otherwise. The travel times for truck and drone matrix satisfies the triangle inequality $\tau_{i,k}^{T} + \tau_{k,j}^{T} \geq \tau_{i,j}^{T}$.

A fleet of K homogeneous trucks, defined as a set of K = {1, 2, 3, 4,..k}, with capacity Q is located at the depot. The maximum number of KD homogeneous drones, defined as a set of KD = {1, 2, 3, 4,..kd}, with the capacity Qd is attached along with each truck. The total length of a drone route in each launch does not exceed a preset limit B (Battery life). In our model, the fleet size of trucks and the number of drones in each truck are given a priori. Denote the set of customer nodes by $C = \{1, 2, 3, 4, 5, 6, \dots, n\}$. As for the depot, we assign it to two unique node numbers at $0(s)$, the starting depot, and $0(r)$, the ending depot. Set $C_0 = C \cup \{0(s)\}$ as the set of customer nodes including the starting depot and set $C_+ = C \cup \{0(r)\}$ as the set of customer nodes including the ending depot. Each customer i (i = 1, . . . , n) is associated with a known nonnegative demand, d_i , to be delivered, and the depot has a fictitious demand $d_o = 0$. The customer demand can be satisfied by either truck or drone delivery.

We define the following decision variables: Let $x_{i,j}^{k}$ equal to 1 if a truck *k* travels along the arc *(i, j)* ∈ *E* and 0 otherwise. This refers to the situation when the truck travels from node $i \in C_0$ to $j \in C_+$ where *i≠j.* Let $y_{i,j}^{kd,k}$ equal to 1 if a drone *kd* of truck *k* travels along the arc *(i, j)* ∈ *E* and 0 otherwise. This refers to the situation when a drone is launched from node $i \in C$ to node $j \in C$ (making a delivery at a customer node) or when a drone is launched from node $i \in C$ to merge with a truck at node $j \in C$ (recharging/swapping a battery at the truck). Let Yt_i^{k} equal to 1 a truck *k* serves customer node *i* and 0 otherwise. Similarly, let $Yd_i^{kd,k}$ equal to 1 if a drone *kd* of truck *k* serves customer node *i* and 0 otherwise.

We denote the variable Tt_j^k as the truck *k* arrival time at node $j \in C_+$ and $Dt_j^{kd,k}$ as the drone *kd* of truck *k* arrival time at node $j \in C_+$. Tt_j^k and $Dt_j^{kd,k}$ are the arrival times of the truck and drones at node *j* respectively. We also define variable $Bat_i^{kd,k}$ to keep track of the battery consumption of a drone *kd* of truck *k* at node *i*. $Bat_i^{kd,k}$ will be reset to zero whenever a drone returns to a truck. Similarly, a variable $Load_i^{kd,k}$ is defined to specify the total load (including that at node *i*) serviced by drone *kd* of truck *k* by the time it reaches customer node i. Lastly, we define other the auxiliary decision variables including 1.) u_i^k which is used in the VRP sub tour elimination constraints (Desrochers & Laporte, 1991), 2.) $la_i^{kd,k}$ which is used to indicate status whether a drone *kd* of truck *k* can be launched from node *i* or not, and 3.) $Zd_1 \& Zd_2$ which is used to indicate whether there is any drone arc coming out from node *i* and entering node *i* accordingly. All the mentioned notations can be summarized as follows.

Indices

i, j	Represent customers, and depot index
k	Represent truck index
kd	Represent drone index

Sets

C	Set of customers, $\{1,\ 2,\ 3,\ 4,\ 5,\ 6, \dots, n\}$
C_0	Set of customer nodes including the starting depot, $C \cup \{0(s)\}$
C_+	Set of customer nodes including the ending depot, $C \cup \{0(r)\}$
D	Set of demands for all customers, $\{d_1, d_2, d_3, \dots, d_n\}$
K	Set of trucks, {1,2,3,…,k}
KD	Set of drones, {1,2,3,…,kd}

Parameters

$\tau^T_{i,j}$	Truck travel time between nodes *i* and *j*
$\tau^D_{i,j}$	Drone travel time between nodes *i* and *j*
n	Number of total customers to be served
Q	Truck capacity (Same for all trucks)
Qd	Drone capacity (Same for all drones)
B	Battery limit (Drone's battery life)
d_i	Customer demand at each node *i*

Variables

z	The sum of all trucks arrival times at the depot
$x^k_{i,j}$	1 if a truck *k* traverses arc (i, j) from customer *i* to customer *j*; otherwise, 0
$y^{kd,k}_{i,j}$	1 if a drone *kd* of truck *k* traverses arc (i, j) from customer *i* to customer *j*; otherwise, 0
Yt^k_i	1 if a truck *k* serves customer node *i* ; otherwise, 0
$Yd^{kd,k}_i$	1 if a drone *kd* of truck *k* serves customer node *i* ; otherwise, 0
Tt^k_j	Truck *k* arrival time at node *j*
$Dt^{kd,k}_j$	Drone *kd* of truck *k* arrival time at node *j*
$Bat^{kd,k}_i$	The battery consumption of a drone *kd* of truck *k* at node *i*
$Load^{kd,k}_i$	The amount of load that a drone *kd* of truck *k* carries
$la^{kd,k}_i$	The state of node *i* which can launch a drone *kd* of truck *k* (0 if launchable state, 1 unlaunchable state)
$zd1_i$	1 if there is a drone arc coming out from node *i;* otherwise, 0
$zd2_i$	1 if there is a drone arc entering node *i;* otherwise, 0
u^k_i	Auxiliary variable for VRP subtour elimination constraints

The proposed MIP formulation of 2EVRPD is presented as follows.

Objective

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

The objective function (1) minimizes the total truck arrival time of trucks at the depot.

Subject to

$$z = \sum_{k \in K} Tt_{0(r)}^{k} \tag{2}$$

Constraint (2) straightforwardly represents the sum of each truck arrival time at the depot.

$$\sum_{k \in K} \sum_{kd \in KD} Yd_i^{kd,k} + \sum_{k \in K} Yt_i^k = 1;\ \forall i \in C \tag{3}$$

Constraints (3) ensure that each customer will receive the package either by a drone or truck. It restricts each customer to be visited exactly once by exactly one vehicle.

$$\sum_{i \in C_+} x_{0(s),i}^{k} = 1;\ \forall k \in K \tag{4}$$

$$\sum_{i \in C_0} x_{i,0(r)}^{k} = 1;\ \forall k \in K \tag{5}$$

Constraints (4) and constraints (5) impose that each truck must depart from and arrive at the depot.

$$\sum_{j \in C_+} x_{i,j}^{k} = \sum_{j \in C_0} x_{j,i}^{k} = Yt_i^k;\ \forall i \in C,\ \ \forall k \in K \tag{6}$$

Constraints (6) ensure the flow conservation of the truck route at each node *i,* which guarantees that whenever the truck *k* arrives at a node, it must depart from the node as well.

$$Yd_j^{kd,k}\left(\sum_{i\in C} y_{j,i}^{kd,k} - 1\right) = 0; \forall j \in C,\ \forall k \in K, \forall kd \in KD \tag{7}$$

$$Yd_j^{kd,k}\left(\sum_{i\in C} y_{i,j}^{kd,k} - 1\right) = 0; \forall j \in C, \forall k \in K, \forall kd \in KD \tag{8}$$

Constraints (7) and constraints (8) define that if a drone *kd* of truck *k* makes a delivery at node *j*, there must be exactly one arc from this assigned drone entering node *j* and exactly one arc from this drone departing node *j*. These sets of constraints ensure the flow conservation of a drone arc at the customer node *j*.

$$\sum_{kd\in KD}\sum_{k\in K}\sum_{j\in C} y_{i,j}^{kd,k} \le 1;\ \forall i \in C \tag{9}$$

$$\sum_{kd\in KD}\sum_{k\in K}\sum_{j\in C} y_{j,i}^{kd,k} \le 1;\ \forall i \in C \tag{10}$$

Constraints (9) and (10) ensure that only a single drone is allowed to travel from and to node *i*. Without these sets of constraints, it can occur the scenario where two drones travel from and to the node *i*.

$$\sum_{kd\in KD}\sum_{k\in K}\sum_{j\in C} y_{i,j}^{kd,k}\,(zd1_i - 1) = 0;\ \forall i \in C \tag{11}$$

$$\sum_{kd\in KD}\sum_{k\in K}\sum_{j\in C} y_{j,i}^{kd,k}\,(zd2_i - 1) = 0;\ \forall i \in C \tag{12}$$

Constraints (11) impose that if a drone departs from node *i,* the auxiliary variable $zd1_i$ must equal to 1. Similarly, constraints (12) impose that if a drone enters node *i,* the auxiliary variable $zd2_i$ must equal to 1 as well.

$$\sum_{j \in C} \sum_{kd \in KD} y_{i,j}^{kd,k} \geq 1 - M\left(2 - Yt_i^k - zd1_i\right);\ \forall i \in C, \forall k \in K \tag{13}$$

$$\sum_{j \in C} \sum_{kd \in KD} y_{j,i}^{kd,k} \geq 1 - M\left(2 - Yt_i^k - zd2_i\right);\ \forall i \in C, \forall k \in K \tag{14}$$

Continuing from constraints (11) and (12), constraints (13) impose that if the customer is served by truck at node *i,* and there is a drone arc leaving node *i,* node *i* is categorized as a *launching* node, where a drone is launched from. Constraints (14) impose that if the customer is served by truck at node *i,* and there is a drone arc entering node *i,* node *i* is categorized as a *landing* node, where a drone lands at. While constraints (7) and (8) preserve the flow conservation of customer node *i* that is served by a drone, constraints (11) to (14) ensure that a drone can land and be launched from a truck which represents the scenario which a drone merges with a truck to retrieve the new load and get a battery swapped.

$$y_{i,j}^{kd,k} \leq 2 - \left(Yt_i^k + Yt_j^k\right);\ \forall i,j \in C,\ \forall k \in K,\ \forall kd \in KD \tag{15}$$

Constraints (15) ensure that a drone can not to travel directly from node *i* to node *j* where node *i* and node *j* are already served by truck.

The sets of constraints (16) to (23) guarantee the correct order of launching and landing operations. A drone can only be launched if it has never be launched before or was previously launched and already returned to receive a service at the truck. The constraints ensure that the same drone can not be launched from the same truck if it was previously launched and have not returned yet.

$$la_i^{kd,k}\left(\sum_{j\in C} y_{j,i}^{kd,k}\right) = 0; \forall i \in N, \forall k \in K, \forall kd \in KD \tag{16}$$

$$la_i^{kd,k}\left(\sum_{j\in C} y_{i,j}^{kd,k}\right) = 0; \forall i \in N,\ \forall k \in K, \forall kd \in KD \tag{17}$$

Constraints (16) and (17) impose that a drone is not allowed to be launched or land at the node *i* once the auxiliary variable $la_i^{kd,k}$ is equal to 1 and vice versa.

$$la_j^{kd,k} \geq 1 - M\left(2 - x_{i,j}^k - \sum_{\substack{q\in C\\ q\neq j}} y_{i,q}^{kd,k} + la_i^{kd,k} + \sum_{\substack{s\in C\\ s\neq i}} y_{s,j}^{kd,k}\right); \forall i, \forall j \in C,\ \forall k \in K, \forall kd\ \in KD \tag{18}$$

$$la_j^{kd,k} \leq 1 + M\left(2 - x_{i,j}^k - \sum_{\substack{q\in C\\ q\neq j}} y_{i,q}^{kd,k} + la_i^{kd,k} + \sum_{\substack{s\in C\\ s\neq i}} y_{s,j}^{kd,k}\right); \forall i, \forall j \in C,\ \forall k \in K, \forall kd\ \in KD \tag{19}$$

$$la_j^{kd,k} \geq 1 - M\left(2 - x_{i,j}^k - la_i^{kd,k} + \sum_{\substack{s\in C\\ s\neq i}} y_{s,j}^{kd,k}\right);\ ; \forall i, \forall j \in C,\ \forall k \in K, \forall kd \in KD \tag{20}$$

$$la_j^{kd,k} \leq 1 + M\left(2 - x_{i,j}^k - la_i^{kd,k} + \sum_{\substack{s\in C\\ s\neq i}} y_{s,j}^{kd,k}\right);\ ; \forall i, \forall j \in C,\ \forall k \in K, \forall kd \in KD \tag{21}$$

Constraints (18) to (21) ensure that if the drone is launched from node *i* and has not returned at node *j,* then the auxiliary variable $la_j^{kd,k}$must be equal to 1, the state which no arc drone leaves or enters node *j*. Constraints (18) and (19) deal with the case that the drone is launched from node *i,* and the truck travels from node *i* to node *j* at which the drone has not yet returned. Constraints (20) and (21) deal with the case when the drone was previously launched (not able to be launched at node *i* again) and has not returned to the node *j* where the truck is making a delivery at yet.

$$la_j^{kd,k} \geq -M\left(2 - x_{i,j}^k - \sum_{\substack{s\in C\\ s\neq i}} y_{s,j}^{kd,k}\right); \ \forall i, \forall j \in C, \ \forall k \in K, \forall kd \in KD \tag{22}$$

$$la_j^{kd,k} \leq +M\left(2 - x_{i,j}^k - \sum_{\substack{s\in C\\ s\neq i}} y_{s,j}^{kd,k}\right); \ \forall i, \forall j \in C, \ \forall k \in K, \forall kd \in KD \tag{23}$$

Constraints (22) to (23) ensure that if the drone returns to node *j* where a truck *k* serves its customer, then the auxiliary variable $la_j^{kd,k}$must be equal to 0, the state which an arc drone can leave or enter the node *j*.

$$Load_j^{kd,k} \geq Load_i^{kd,k} + D_j - M\left(1 - y_{i,j}^{kd,k} + Yt_j^k\right); \ \forall i, \forall j \in C, \ \forall k \in K, \forall kd \in KD \tag{24}$$

$$Load_j^{kd,k} \leq Load_i^{kd,k} + D_j + M\left(1 - y_{i,j}^{kd,k} + Yt_j^k\right); \ \forall i, \forall j \in C, \ \forall k \in K, \forall kd \in KD \tag{25}$$

$$Load_i^{kd,k} \leq Qd; \forall i \in N, \ \forall k \in K, \forall kd \in KD \tag{26}$$

The sets of constraints (24) to (25) deal with the load of each drone to ensure that the amount of drone's load is updated whenever a drone makes a delivery at a customer node *j*. The load will be reset to zero when a drone flies back to receive service or a battery swap at any truck customer node. Constraints (26) require that the load must be less than the drone's capacity (*Qd*). We initialize the load for each drone at each truck to be equal to zero at the depot.

$$Bat_j^{kd,k} \geq Bat_i^{kd,k} + \tau_{i,j}^D - M\left(2 - y_{i,j}^{kd,k} - Yd_i^{kd,k}\right); \ \forall i, \forall j \in C, \forall k \in K, \forall kd \in KD \tag{27}$$

$$Bat_j^{kd,k} \leq Bat_i^{kd,k} + \tau_{i,j}^D + M\left(2 - y_{i,j}^{kd,k} - Yd_i^{kd,k}\right); \ \forall i, \forall j \in C, \forall k \in K, \forall kd \in KD \tag{28}$$

Constraints (27) and (28) update the drone's battery consumption whenever a drone travels from a drone customer node *i* and make a delivery at a customer node *j*.

$$Bat_j^{kd,k} \geq \tau_{i,j}^D - M\left(2 - y_{i,j}^{kd,k} - Yt_i^k\right);\ \forall i, \forall j \in C, \forall k \in K, \forall kd \in KD \tag{29}$$

$$Bat_j^{kd,k} \leq \tau_{i,j}^D + M\left(2 - y_{i,j}^{kd,k} - Yt_i^k\right);\ \forall i, \forall j \in C, \forall k \in K, \forall kd \in KD \tag{30}$$

Similarly, constraints (29) and (30) keep track of the drone's battery consumption in the case when the drone is initially launched from the truck.

$$Bat_i^{kd,k} \leq B; \forall i \in N,\ \forall k \in K, \forall kd \in KD \tag{31}$$

Constraints (31) ensure that the amount of battery consumption of each drone must be less than its drone's battery limit at any point in time.

$$\sum_{i \in C} D_i \left(Yt_i^k\right) + \sum_{i \in C} \sum_{kd \in KD} D_i(Yd_i^{kd,k}) \leq Q; \forall k \in K \tag{32}$$

Constraints (32) enforce that the total delivery loads of both truck and drone combined must be less than the truck capacity at each truck route *k*.

$$\sum_{j \in C} y_{i,j}^{kd,k} \left(Tt_i^k - Dt_i^{kd,k}\right) = 0;\ \forall i \in C_0, \forall k \in K, \forall kd \in KD \tag{33}$$

$$\sum_{j \in C} y_{j,i}^{kd,k} \left(Tt_i^k - Dt_i^{kd,k}\right) = 0;\ \forall i \in C, \forall k \in K, \forall kd \in KD \tag{34}$$

Constraints (33) state that the departure time of drones and trucks must be the same. Also, once the drone and truck are in the same node, they must wait for each other before each of them can leave the node (Murray and Chu, 2015). Similarly, constraints (34) ensure that the arrival time of both truck and drone will be the same when they merge at the same node. These sets of constraints are based on the assumption that if either the drone or truck arrives earlier than the other, the earlier one has to wait until the later one arrives (both constraints are binding, resulting in the same arrival time of both truck and drone).

$$Tt_j^k \geq Tt_i^k + \tau_{i,j}^T - M\left(1 - x_{i,j}^k\right);\ \forall i \in C_0, \forall j \in C_+, \forall k \in K \tag{35}$$

$$Dt_j^{kd,k} \geq Dt_i^{kd,k} + \tau_{i,j}^D - M\left(1 - y_{i,j}^{kd,k}\right); \forall i, \forall j \in C, \forall k \in K, \forall kd \in KD \tag{36}$$

Constraints (35) keep track of the arrival time of the truck at every node. It adds the truck travel time to the previous customer node when the truck travels from one customer node to another customer node. Similarly, constraints (36) keep track of the arrival time of the drone at the node to which the drone returns after making a delivery.

$$u_i^k - u_j^k + Q(x_{i,j}^k) \leq Q - D_j;\ \forall i,\ \forall j \in C \cup C_0 \cup C_+, \forall k \in K \tag{37}$$

$$D_i \leq u_i^k \leq Q\ ;\ \forall i,\ \forall j \in C \cup C_0 \cup C_+, \forall k \in K \tag{38}$$

Constraints (37) and (38) are sets of the Desrochers and Laporte (DL) sub tour elimination constraint which ensures that there is no sub tour in all tours of the trucks (Desrochers & Laporte, 1991).

At Initial State (Time = 0)

$$Bat_i^{kd,k} = 0,\ Load_i^{kd,k} = 0, Tt_i^k = 0,\ Dt_j^{kd,k} = 0\ \forall i \in N, \forall j \in N, \forall k \in K, \forall kd \in KD \tag{39}$$

The set of constraints (39) set the initial departure time of drones and trucks as well as the battery consumption and load at time zero to be zero for all nodes *i*, all drones *kd* and all trucks *k*.

$$x_{i,j}^k\ \in\ \{0,1\}\quad \forall i, \mathrm{j}\ \in C \cup C_0 \cup C_+ \tag{40}$$

$$y_{i,j}^{kd,k} \in \{0,1\}\quad \forall i, \mathrm{j}, \mathrm{k}\ \in C \tag{41}$$

$$Yt_i^k \in \{0,1\}\,, Yd_i^{kd,k} \in \{0,1\}\ \forall i \in C, \forall k \in K, \forall kd \in KD \tag{42}$$

$$Tt_j^k \geq 0, Dt_j^{kd,k} \geq 0, Bat_i^{kd,k} \geq 0,\ Load_i^{kd,k} \geq 0\quad \forall i \in C \cup C_0 \cup C_+, \forall k \in K, \forall kd \in KD \tag{43}$$

$$la_i^{kd,k} \in \{0,1\},\ zd1_i \in \{0,1\}, zd2_i \in \{0,1\}\ \ \forall i \in C, \forall k \in K, \forall kd \in KD \tag{44}$$

Constraints (40–44) specify the types and ranges of the variables. Note that the M value must be large enough. Thus, we can use the total time of all the delivery routes made by trucks alone. i.e., solve a regular CVRP.

4.4 Solution Methods

Mixed Integer Programming (MIP) formulations (1) through (44) for 2EVRPD have been developed and presented in the previous sections. Although we are able to obtain the exact optimal solutions to small-size problems (less than 10 nodes) with the CPLEX solver, solving the larger-size problems could take too much time to obtain the exact solution. In this section, we describe how to solve the defined problem within the practical time frame using two proposed heuristics: Large Neighborhood Search (LNS) and Drone Truck Route Construction (DTRC).

4.4.1 Drone Truck Route Construction Heuristic

Drone Truck Route Construction is considered as a constructive heuristic which gradually builds a feasible solution while keeping an eye on solution cost. The structure of an algorithm composes of two phases. The first phase includes the process of building a solution for the Capacitated VRP using fast and efficient classical heuristics. The second phase is to construct a feasible drone sub route within the truck main route while maintaining the feasibility constraints such as battery life, capacity and the order of launching and landing operation. We intend to implement Drone Truck Route Construction (DTRC) to obtain the solution for 2EVRPD using short computational time. Algorithm DTRC shows the basic structure of the proposed method.

Input parameters

Trucks, Drones, All Nodes, Demand, Travel Time, Capacity, Battery

Variables

S^{CVRP}: Solution of Capacitated Vehicle Routing Problem

S^{2EVRPD}: Solution of Two Echelon Vehicle Routing with Drones

Route: The truck delivery path as well as drone delivery path

RemainN: This variable keeps track of the unvisited nodes

LandingN: This variable keeps track of the landing nodes

Available Drones: The set of available drones that can be selected for launching

Drone: The selected drone from *Available Drones*

Next Drone Delivery Node: The selected customer node to be served by drone

Next Truck Delivery Node: The selected customer node to be served by truck

Drone Bat: The cumulative amount of battery that a drone consumes

Drone Load: The cumulative amount of load that a drone carries

ϑ: Ratio of $\frac{\mathcal{D}}{\mathcal{W}}$; where $\mathcal{D}$ is the distance between the current drone node and an unvisited node
$\mathcal{W}$ is the current drone load plus the demand of an unvisited node

4.4.1.1 CVRP Solution

The Capacitated Vehicle Routing Problem (CVRP) is a Vehicle Routing Problem with additional constraints on the capacities of the vehicles. In a CVRP, each location has a demand, such as weight or volume, corresponding to the item to be picked up or delivered there. In addition, each vehicle has a maximum capacity for the total quantity it can carry. To quickly obtain the CVRP solution, we adopt the Clarke and Wright Saving Algorithm (Clarke and Wright, 1964), which is by far the best-known approach and yet conceptually simple yields reasonably good solutions to the CVRP problem (Laporte, 2009).

Algorithm: DTRC.

Input: *Trucks, Drones, All Nodes, Demand, Travel Time, Capacity, Battery*

Output: S^{2EVRPD}

1. Generate S^{CVRP} using *Savings (Truck, Nodes, Demand, Travel Time)*

2. **For** *Route* $\in S^{\text{CVRP}}$

3. Set *RemainN* = *{All nodes in Route}*

4. Initialize *1st Launch Node = 1st Truck Node* in *Route, LandingN = [];*

5. **While** *RemainN* $\notin \emptyset$

6. Select *Drone* from *Available Drones* = {1,2,…kd}

7. **If** *Available Drones* == $\emptyset$, proceed to line (17), **else**

8. Update *Available Drones* = *Available Drones* \ *Drone*

9. **Repeat**

10. Choose *Next Drone Delivery Node* ← min ϑ

11. Update *Drone Bat, Drone Load, Route*

12. Update *RemainN* = *RemainN* – *{Next Drone Delivery Node}*

13. **Until** *Drone Load* ≥ Drone Capacity || *Drone Bat* ≥ Battery

14. Update *LandingN* = *LandingN* ∪ *Last Drone Delivery Node*

15. **End** Repeat

16. **End** If

17. Choose *Next Truck Delivery Node* ← min $\mathcal{D}$ from *RemainN* ∪ *LandingN*

18. Update *Route, RemainN* = *RemainN* - *Next Truck Delivery Node*

19. **If** *Next Truck Delivery Node* == *LandingN*

20. Reset $Drone\ Load_{Drone}$, $Drone\ Bat_{Drone}$ = 0

21. *Available Drones* = *Available Drones* ∪ *Drone*

22. **End** If

23. **End** while

24. Return S^{2EVRPD} for each *Route*

25. **End** For

26. **Return** S^{2EVRPD}

The Saving algorithm uses the following steps:

1. Create n truck routes (0, i ,0) for i = 1,…, n
2. Compute the savings $s_{i,j}$= $c_{i,0} + c_{0,j} - c_{i,j}$ for i, j = 1,…, n.
3. List the savings in the descending order
4. Starting from the top of the savings list, merge two routes associated with the largest savings, given that:

   Two delivery nodes are not in the same route

   The two nodes must directly connect to the depot, e.g. $(0, j)$ and $(i, 0)$

   The total demand of the merged route must not exceed the truck capacity
5. Repeat step (4) until no savings can be used.

After obtaining the CVRP solution from the Saving Algorithm, we apply some well-known local search such as, 2 opt, simple relocate and swap move (Gendreau, 2008), to improve the solution quality. This completes the first phase of the algorithm (Line 1.)

4.4.1.2 Drone Route Construction

Using the CVRP solution obtained from the Saving Algorithm, we gradually build up the new truck routes and drone sub-routes to generate a feasible 2EVRPD solution. The heuristic would eliminate a non-feasible solution while constructing the route during this phase. For each truck route, we implement a famous modified Nearest Neighbor (NN) Algorithm to build a complete drone route. The algorithm constructs multiple sub drone routes within each of the truck route until no more drone route can be constructed. The Drone Route Construction steps can be provided in details as follows:

Step 1: For each truck route obtained from the CVRP solution, select the initial truck node where the drone can be launched. The initial node is usually the node that is located closest to the depot (Line 2-4).

Step 2: Select one of the available drones to be launched and then select one of the truck nodes as a drone delivery node. Unlike the traditional NN which selects the next visited node based on the ranking of the distance between the current node and an unvisited node, we rank the next unvisited node using the ratio $\vartheta = \frac{\mathcal{D}}{\mathcal{W}}$ where $\mathcal{D}$ is the distance between the current drone node

$$\mathcal{D} = \| Node_{current} - Node_i \| \text{ ; } \forall i \in Unvisted \text{ Nodes}$$

and an unvisited node, and $\mathcal{W}$ is the current drone load plus the demand of an unvisited node.

$$\mathcal{W} = Load_{current} + d_i \text{ ; } \forall i \in Unvisted \text{ Nodes}$$

The node with the lowest ϑ (highest rank) will be selected as the drone delivery node. We use this ratio because it is intuitively more efficient for a drone to drop a heavy package at a closer distance to preserve more energy, which should be prioritized first (Line 10).

Step 3: Remove the selected drone delivery node from the truck route and add it to the drone route. Update the drone's battery consumption and load (Line 11,12).

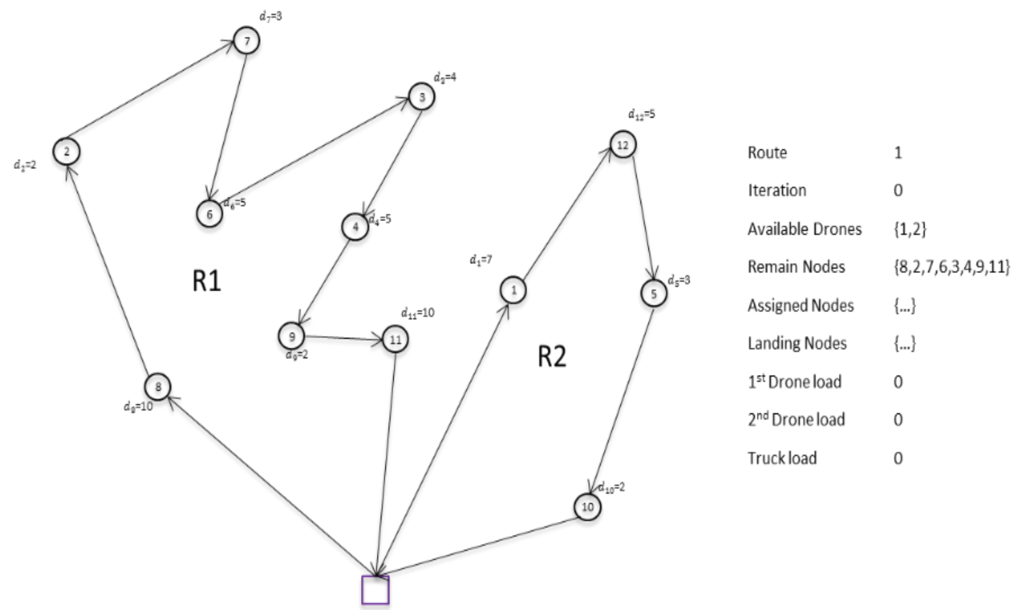
Step 4: Repeat steps 2 & 3 for the same drone until no other nodes can be selected due to either 1.) the drone's load exceeds its capacity or 2.) drone's energy consumption exceeds its capacity. The last customer node in a drone route is called "landing node".

Step 5: At the current truck node, select the next available customer node for a next truck delivery from 1.) the remaining nodes that have not been previously selected to be in the drone route yet or 2.) the landing node from any drone route. If the latter is selected, we obtain a complete drone route and this drone will be available for the new selection in the next iteration (Line 19-21). Otherwise, we can select the next customer node to be served by a truck and can consider choosing one of the remaining drones to create another drone route. Please note that we simply select the node which has the closest distance to the current truck node (min $\mathcal{D}$).

Step 6: Repeat steps 2 – 5 until no drone can be selected and we have a complete truck route returning back to the depot. Complete the same steps for all the routes in the CVRP solution.

Figure 4.2a.) to 4.2d.) represent the graphical examples of steps 1-6 on phase 2 of DTRC heuristic.

a.)



b.)

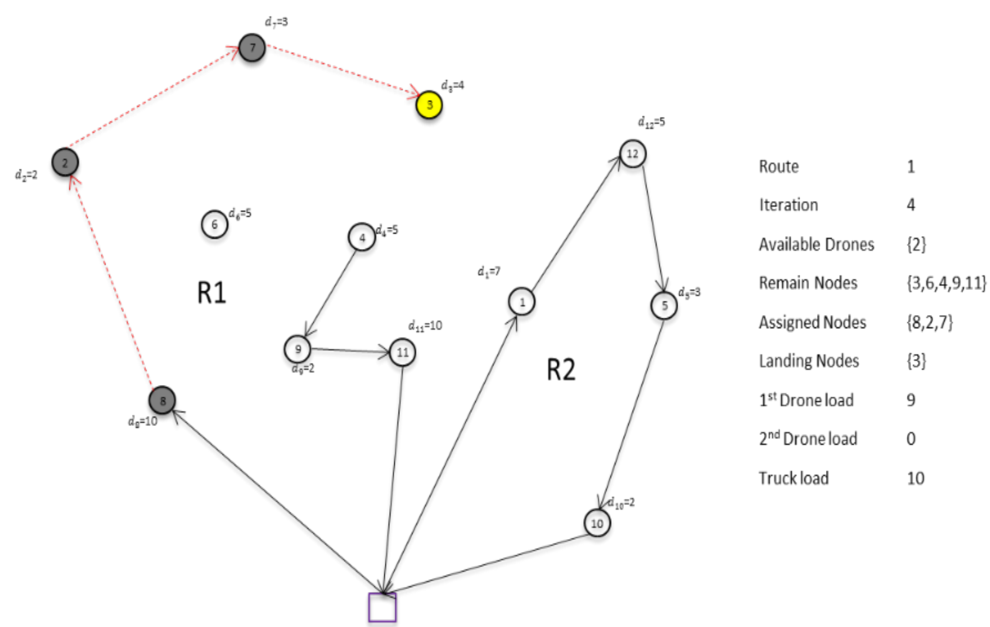
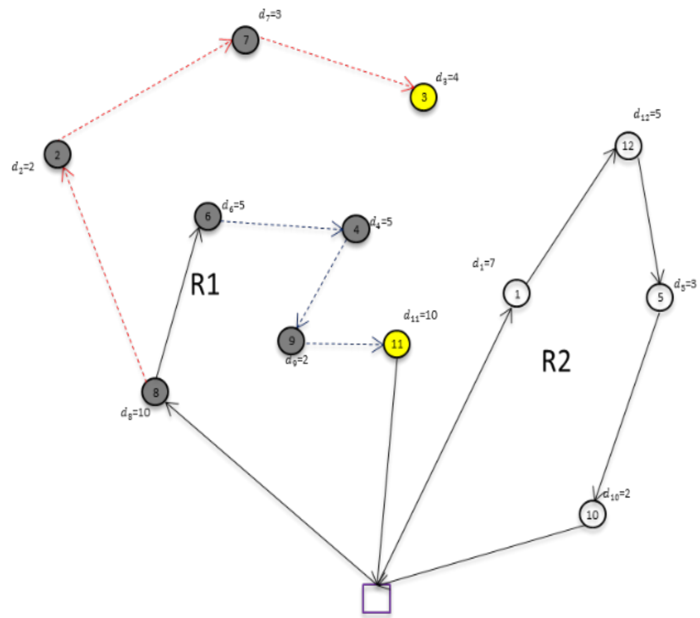


Figure 4.2 Example of the phase 2 of DTRC algorithm. a.) Illustration of Initial CVRP solution, b.) Illustration of 4th iteration in phase 2 of DTRC, c.) Illustration of 8th iteration in phase 2 of DTRC d.) Illustration of 10th iteration in phase 2 of DTRC

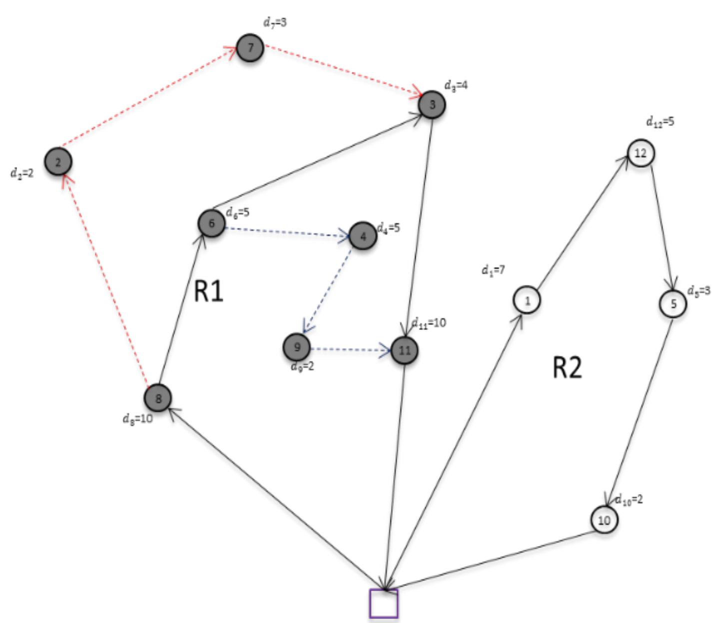
Figure 4.2 continued

c.)



Route	1
Iteration	8
Available Drones	{...}
Remain Nodes	{3,11}
Assigned Nodes	{8,2,7,6,4,9}
Landing Nodes	{3,11}
1st Drone load	9
2nd Drone load	7
Truck load	15

d.)



Route	1
Iteration	10
Available Drones	{1,2}
Remain Nodes	{...}
Assigned Nodes	{8,2,7,6,4,9,3,11}
Landing Nodes	{...}
1st Drone load	0
2nd Drone load	0
Truck load	29

4.4.2 Large Neighborhood Search

The proposed metaheuristic follows the basic structure of the Large Neighborhood Search proposed by Paul Shaw in 1998 (Shaw, 1998). LNS is based on a process of continual relaxation and re-optimization. An initial feasible solution of the problem is destroyed and repaired iteratively to gradually improve the solution quality. For the typical VRP, the solution is relaxed when nodes are removed from the solution routes and the solution is re-optimized by reinserting the nodes back into the routing solution. The framework of LNS has recently been used to successfully solve multiple variants of Vehicle Routing Problems (Pisinger and Ropke, 2010). In a complex problem like VRP, the classical algorithm makes it quite difficult to walk away from the local optimal solution and provides a limited search space. LNS offers a large move that could expand the solution search space by disintegrating a large part of the previous solution and give the freedom to create a new one (Schrimpf et al., 2000). Since our proposed model has many side constraints such as, order of launching and landing operation as well as time adjustment for both truck and drone, it would be more applicable to implement LNS to our problem rather than using standard local search move operators. Algorithm LNS-2EVRPD shows the basic structure of the proposed LNS for 2EVRPD.

Algorithm: LNS-2EVRPD

Input: *Trucks, Drones, All Nodes, Demand, Travel Time, Capacity, Battery*

Output: S^{Best}

1. Generate $S^{Best} \leftarrow S^{Current} \leftarrow$ *DTRC (Truck, Drones, Nodes, Demand, Travel Time)*
2. Initialize $time = 0$, $i = 0$
3. **Repeat**
4. **While** $i \leq i_{max}$
5. Enter *Destroy Phase*
6. $S^{TempD1} \leftarrow$ Call *Drone node removal* ($S^{Current}$)
7. $S^{TempD2} \leftarrow$ Call *Truck node removal* (S^{TempD1})
8. $S^{TempD3} \leftarrow$ Call *Sub-drone route removal* (S^{TempD2})
9. Enter *Repair Phase*
10. $S^{TempR1} \leftarrow$ Call *Drone node insertion* (S^{TempD3})
11. $S^{TempR2} \leftarrow$ Call *Truck node insertion* (S^{TempD3})

12. S^{TempR3} ← Call *Drone route creation* *(*S^{TempD3}*)*
13. Select S^{Temp} = min *(*S^{TempR1}, S^{TempR2}, S^{TempR3}*)*
14. **If** Cost *(*S^{Temp}*)* < Cost *(*$S^{Current}$*)*
15. $S^{Current} \leftarrow S^{Temp}$
16. $i = 0$
17. **Else**
18. $i = i + 1$
19. **End** If
20. **End** While
21. **If** Cost *(*$S^{Current}$*)* < Cost *(*S^{Best}*)*
22. $S^{Best} \leftarrow S^{Current}$
23. $i = 0$
24. **Else**
22. $S^{Current}$ ← *DTRC (Truck, Drones, Nodes, Demand, Travel Time)*
23. $i = 0$
24. **End** If
26. **Until** $time \geq time_{max}$
26. **Return** S^{Best}

The proposed algorithm starts by generating an initial solution from all inputs previously mentioned in the earlier section. This initial solution will be stored as a current solution as well as the global best solution (Line 1). At each iteration, a partial solution destruction is performed sequentially using three destroy operators on the current solution routes, then the destroyed solution is repaired by the repair operators (Line 6 – Line 12). The best repair solution will be selected and recorded as the temporary solution (Line 13). If the objective of a temporary solution is lower than the one from the current solution, we accept the new current solution and the index *i* is reset to 0 (Line 14-16). Otherwise, we still keep the old current solution and perform another round of destroy-repair operation to the temporary solution. We keep improving the current solution until the index i reaches i_{max}. Then, we compare the objective between the current solution and the global best solution. If the current solution provides a lower objective value, we accept and update the new global best solution (Line 21-23). If there is no objective improvement after a large number of iterations, then the algorithm will restart from a new initial solution (Line

22). We repeat the same steps until the $time \geq time_{max}$ and return the best solution. All different types of destroy operators and repair operators will be described in the next section.

4.4.2.1 Initial Solution

Since the 2EVRPD is a new derived problem, which has the solution structure different from other VRP or 2EVRP solution, we can not find any specific algorithm that could apply to generate a solution for our problem directly. As of now, we generate the initial solution using our Drone Truck Route Construction Heuristic (DTRC) described in the previous section. Due to some randomness in DTRC, the initial solution we generate in each run could increase the area of search space in LNS.

4.4.2.2 Destroy Operators

Our algorithm relies on three different destroy operators which are all invoked at each iteration in sequential order. They are applied to both drone delivery nodes and truck delivery nodes. All of them select different types of nodes from the current solution. When applicable, all random samples are uniformly distributed within their given interval. The operators are presented in the sequential order of execution as follows.

Drone node removal: This operator removes customer nodes who are currently served by drones in the route and add them to the re-insert list. We assign a parameter p_1, which denotes the percentage of drone nodes to be removed. In each iteration, we allow $\lceil p_1 \cdot C_{Only\ Drone} \rceil$ nodes to be removed, with $C_{Only\ Drone}$ being the set of customer nodes who receive deliveries by drones only. Each removed node is selected randomly. The operator accepts two inputs: the current solution route and the set of drone nodes to be destroyed and, return the updated route after removing the drone nodes. If all customer nodes in one of the sub drone routes are removed, the truck customer nodes which previously launched and retrieved a drone will become available for launching and retrieving any available drone.

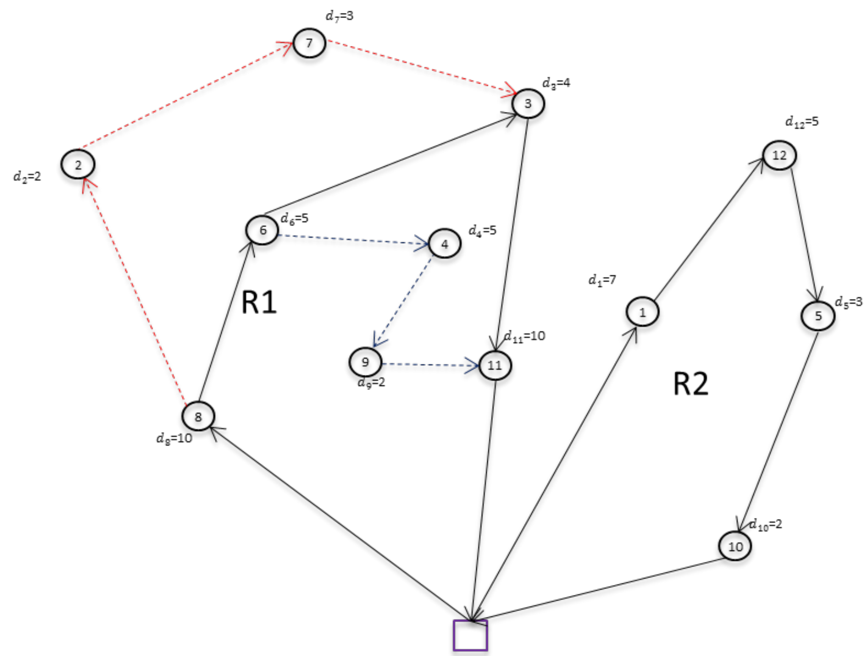
Truck node removal: This operator removes customers who are served by trucks from the current route solution and add them to the re-insert list. This operator, however, does not consider removing the truck node which performs either launching or landing operation since doing so would consequently erase the entire sub drone route associated with the particular truck node.

Please note that we have also created another operator that deals with the sub-drone route removal directly and will be described subsequently. Similar to the *Drone node removal,* we assign a parameter p_2, which denotes the percentage of truck nodes to be removed. In each iteration, we allow $\lceil p_2 \cdot C_{Only\ Truck} \rceil$ nodes to be removed, with $C_{Only\ Truck}$ being the set of customer nodes who receive deliveries by truck only. The operator accepts and returns the same type of output as the previous operator. If all customer nodes are removed from the truck route, the solution for that truck route becomes empty and the truck stays at the depot.

Sub-drone route removal: This operator removes one or multiple sub-drone routes (level two) from the truck route (level one). Each sub-drone route includes of the customer nodes who are currently served by drones and the truck nodes which perform launching and landing operation. We assign a parameter p_3, which denotes the percentage of sub-routes to be removed. In each iteration, we allow $\lceil p_3 \cdot R_{Sub\ Route} \rceil$ routes to be removed, with $R_{Sub\ Route}$ being the set of sub drone routes associated with the selected truck route. All customer nodes in the sub-drone route are added to the re-insert list. If all sub-routes are removed from the main truck route, the remaining customers will be served by truck alone and the solution route is reduced to the level 1 delivery which is equivalent to the solution from regular CVRP.

Figure 4.3a.) represents the current solution and Figure 4.3b.) represents the solution after all three destroy operators are executed.

a.)



b.)

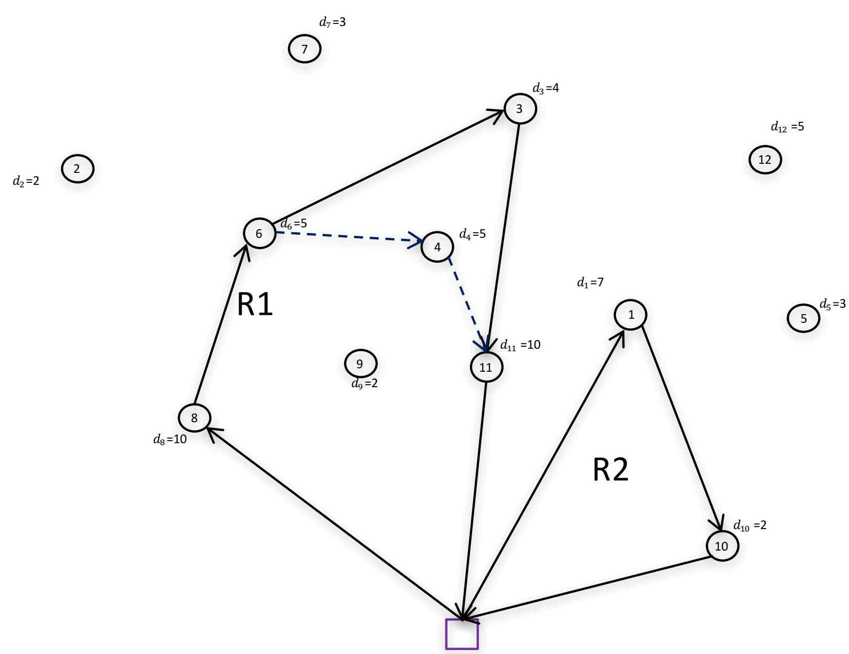


Figure 4.3 Example of using Destroy operators on 2EVRPD solution. a.) Illustration of current 2EVRPD solution, b) Solution after the destroy operators: Drone node removal {9}, Truck node removal {12,5}, Sub-drone route removal {2,7}

4.4.2.3 Repair Operators

After performing all three types of destroy operators sequentially, all the nodes in the re-insert list must be inserted back into the routes of an existing solution. We apply three types of repair operators to all nodes in the re-insert list. At the repair phase, each node would go through all three repair operators in parallel. Each repair operator returns the new routing solution with the selected node already inserted in the route. We compare the solutions obtained from three operators. The best-repaired solution will be selected and become the current solution of the problem. We perform the repair process until no node is left in the re-insert list and reevaluate the current solution with the global solution. The repair operators are described as follows.

Drone node insertion: The operator takes a node input and inserts it into one of the existing drone sub-routes (level 2) in the solution. The route to be inserted to is not restricted to the one that this node is removed from and can be a sub drone route in any truck route. The insertion is achieved with a simplified cheapest insertion heuristic, the method to insert the node into the position with the which gives the lowest increase in total costs. The insertion is prohibited in the drone route if the new drone route after insertion causes 1.) the drone's load to exceed its capacity and 2.) the drone's battery consumption to exceed its battery limitation.

Truck node insertion: The operator works similarly to the *Drone node insertion* by inserting the selected node into one of the truck routes (level 1) at the current solution. The operator searches for all the feasible positions to insert the node and selects the one with the lowest increase in total cost. If the current capacity of trucks for all routes is full, the node can be inserted into an empty route which creates one more truck route in the solution.

Drone route creation: The operator creates a new sub drone route by inserting the selected node between a pair of truck nodes. It is required that no drone arc can enter or leave both truck nodes. In other words, no drone is being launched or landing at the truck nodes. Considering a pair of (i,k) nodes where node i must precede node k, we insert node j between node i and node k to create the new drone sub route (level 2). A drone is launched from node i, makes a delivery at node j and returns to another truck node k. The operator searches for the cheapest pair (i,k) among all possible combinations to construct a new drone route with the lowest increase in total cost. Figure 4.4 represents the solution after all three repair operators are executed.

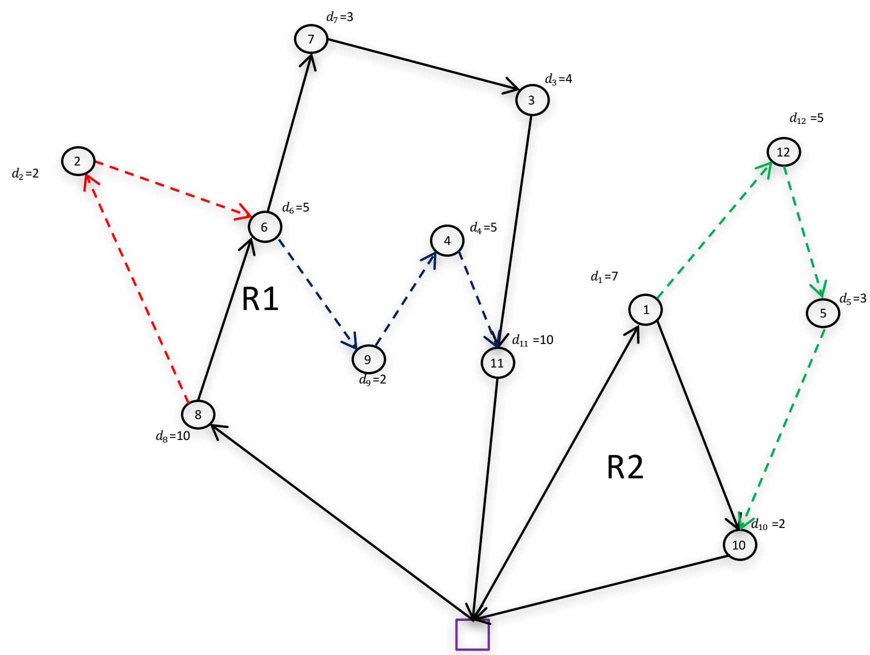


Figure 4.4 Example of using Repair operators on 2EVRPD solution

4.5 Computational Examples and Results

This section examines the formulated MIP problem and the proposed algorithms using numerical examples. Since our problem has some similarities with the classical CVRP with the use of drones on multiple drops delivery, the experiments were done on the CVRP benchmark instances from the literature. We conducted our experiments from different sets of instances, taken from four classical sets of the CVRP benchmark from Augerat et al. (set A, B and P) and Christofides and Elion (set E). The input data are available online at the Capacitated Vehicle Routing Problem Library (Xavier, 2014). We assume that the travel time can be represented by the cost metric associated with the benchmark problem. We set the truck travel time to be 1.5 time units longer than the drone travel time ($\tau^T_{j,i} = 1.5\, \tau^D_{i,j}$) since the drone speed is roughly about 1.5 times faster than the truck speed (Brar et al., 2015). The truck capacity and the number of trucks for each instance are excerpted from the instance input as well. Other parameters are currently assigned randomly and are subject to more calibrations in the future work. For the headers of all the tables presented in this section, we refer n as the number of customer nodes; K as the number of trucks; Kd as the number of drones; Q as the truck's capacity; and Qd as the drone's capacity. We assume that both trucks and drones travel in Euclidean space. All the algorithms were executed in Matlab on a computer with 2.7GHz Intel Core i5 with 8GB ram RAM running Windows 7 64-bit mode.

All the Mixed-Integer Linear Programming models were solved using GAMS 23.51 with CPLEX solver.

4.5.1 Comparison of 2EVRPD-MIP Model and CVRP-MIP Model on Small-Size Problems

This section compares the solution between the proposed 2EVRPD and the classical CVRP on small-size problems. Exact solutions are also obtained for both models using the CPLEX solver for comparison. The objective of this experiment is to evaluate the cost (time) saving by implementing the multiple drops by drones operation with trucks under the small-size problem circumstance. We also want to get an estimation of how long it would take to solve the 2EVRPD using the MIP method to obtain an optimal solution. To simplify and reduce the amount of runtime, we modified the benchmark problems by reducing the number of customer nodes to be 8 and limiting the number of trucks to be 2. The results are shown in Table 4.1 and Figure 4.5.

Table 4.1 Comparison of the results between MIP-2EVRPD and MIP-CVRP on small-size instance

Instance	n	K	Q	Kd	Qd	CPLEX MIP Solver				Improvement (%)
						2EVRPD		CVRP		
						Solution (Optimal)	Runtime (Second)	Solution (Optimal)	Runtime (Second)	
A1-n8-k2	8	2	100	2	40	289	2966.17	338	69.02	**14.50**
A2-n8-k2	8	2	100	2	40	226	2618.37	305	54.73	**25.90**
A3-n8-k2	8	2	100	2	40	177	3600.00	204	71.89	**13.24**
B1-n8-k2	8	2	100	2	40	313	3122.41	340	70.5	**7.94**
B2-n8-k2	8	2	100	2	40	176	3345.11	189	46.38	**6.88**
P1-n8-k2	8	2	100	2	40	114	2612.39	140	41.12	**18.57**
P2-n8-k2	8	2	100	2	40	125	2786.12	148	37.46	**15.54**
						Average	3007.22	-	55.87	**14.65**

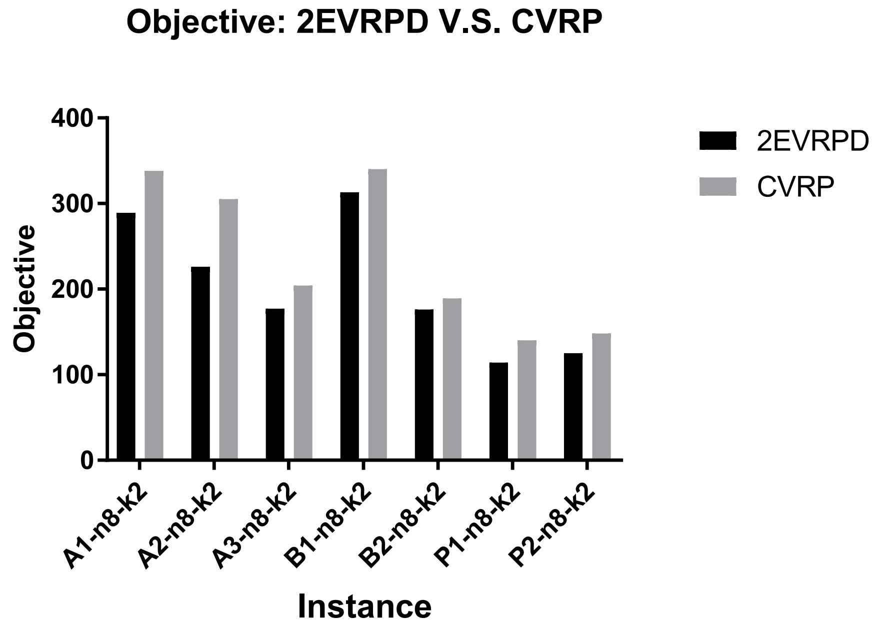


Figure 4.5 The grouped bar chart showing the difference between 2EVRPD and CVRP objective on different instances

The results show that implementing the drones with multiple drops feature could reduce the objective approximately by 14.65% (6.88% to 25.90%) depending on the instance. In general, we expect the saving to be lower than the classical CVRP but varied by the location of the customer nodes. As the solver takes a significant amount of time (as showing 3007.22 seconds on average) to generate the results even for the small-size problem, it might not be worth the time to compute for an exact solution on the larger-size problems. Therefore, we conducted more experiments to solve 2EVRPD using the proposed heuristics described in Section 4.4.

4.5.2 Comparison of the Proposed Heuristics and the 2EVRPD-MIP Model

From Section 4.5.1 experiment, the results indicate that CPLEX can take quite a bit of time before returning an exact solution for the small-size problem. In this section, we tested our proposed heuristics with the larger problem size that CPLEX can not obtain the exact solutions within a reasonable amount of time. We retrieved some of the instances from the benchmark problems and tested them with our heuristics. We ran each heuristic 10 iterations for each instance and reported the best objective among the iterations, the average objectives and the average runtime. We also computed the GAP of the average objective between the one from CPLEX and the other two from both heuristics. The negative GAP indicates that the solutions from heuristic algorithms perform better. In addition, we ran the CPLEX for one hour for each instance and compare the results with the proposed algorithms. The numerical results are listed in Tables 4.2, and the time plot is shown in Figure 4.6.

Table 4.2 Results of the proposed heuristic and CPLEX on the larger instances

Instance	n	K	Q	Kd	Qd	CPLEX		LNS Heuristic				DTRC Heuristic			
						Obj	Time	Best	Average	Avg GAP from MIP (%)	Avg Runtime (Second)	Best	Average	Avg GAP from MIP (%)	Avg Runtime (Second)
A-n32-k5	31	5	100	2	35	1364	3600	675	701.80	**-48.55**	73.17	800	812.9	**-40.40**	0.61
A-n33-k5	32	5	100	2	35	1275	3600	547	566.50	**-55.57**	67.93	656	666.1	**-47.76**	0.67
A-n33-k6	32	6	100	2	35	1055	3600	614	633.60	**-39.94**	72.70	686	697.2	**-33.91**	0.46
B-n31-k5	30	5	100	2	35	1000	3600	649	653.90	**-34.61**	67.33	667	670.7	**-32.93**	0.44
B-n34-k5	33	5	100	2	35	1360	3600	742	751.50	**-44.74**	78.90	771	787.1	**-42.13**	0.62
P-n16-k8	15	8	35	2	20	441	3600	442	442.00	0.23	60.45	467	467	5.90	0.11
P-n19-k2	18	2	160	2	40	215	3600	146	154.50	**-28.14**	61.61	222	228.8	6.42	0.35
P-n20-k2	19	2	160	2	40	242	3600	153	157.90	**-34.75**	63.25	213	215.8	**-10.83**	0.32
P-n21-k2	20	2	160	2	40	246	3600	143	147.50	**-40.04**	62.44	210	231.1	**-6.06**	0.52
P-n22-k2	21	2	160	2	40	265	3600	148	156.90	**-40.79**	64.89	216	216	**-18.49**	0.52
									Average	**-36.69**	67.27			**-22.02**	0.46

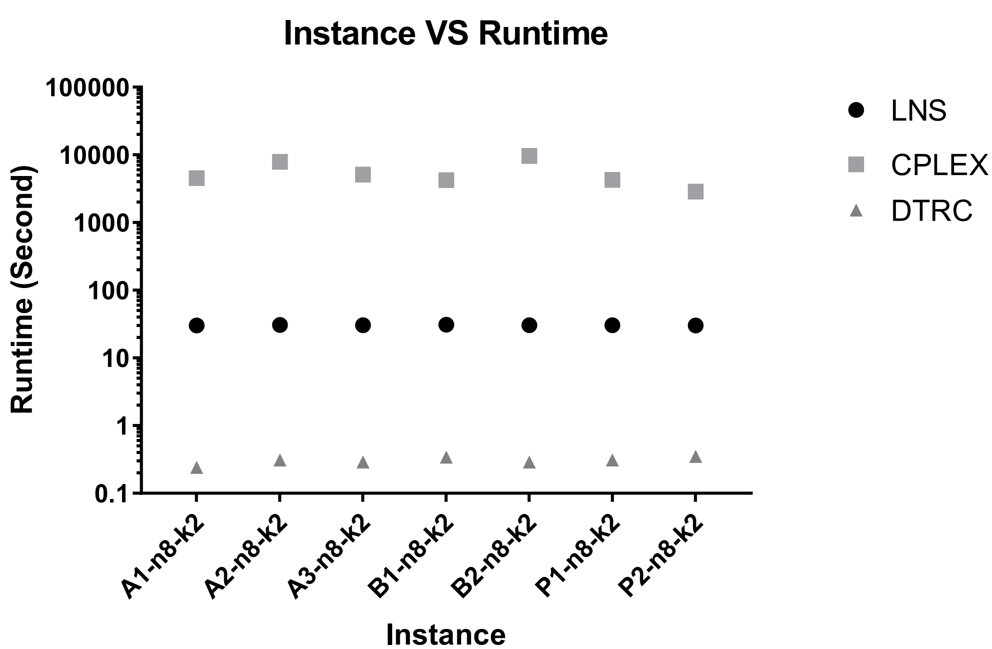


Figure 4.6 The plot showing the runtime of different solution methods

As the results show, both LNS and DTRC algorithms can obtain better results in significantly less time compared to CPLEX. On average, we see -36.69% average GAP from CPLEX for the LNS and -22.02% average GAP from CPLEX for the DTRC. Please note that the objective reported from CPLEX is the current solution found in one hour. The results demonstrate that the two heuristics can return reasonably good quality solutions when comparing to the solution of CPLEX within one hour. In terms of computational time, we plot the run time for each method (Figure 4.7) which shows that DTRC can return the solution using the lowest CPU runtime of only 0.46 seconds followed by LNS with the average of 67.27 seconds and CPLEX with the fixed time at 3600 seconds respectively.

4.5.3 Comparison of the Proposed Heuristics and the Optimal Solution of CVRP on Various Instances

In this section, we want to test the performance of the proposed heuristics on different variations of instances. A total of 50 test benchmark instances were used in this experiment. Each benchmark instance was run 10 times independently. We compare the performance of our two heuristics, LNS and DTRC, with the CVRP optimal solutions for each instance from the literature. The results are shown in Table 4.3.

In terms of solution quality, the LNS returns better objective values than the DTRC's for all instances with an average GAP of about 11% However, the computational time of the DTRC heuristic is about 98% lower on average. Figure 4.7 shows the variable number growth for different problem sizes and the time increase as problem size increases. The LNS plot shows that the number of variables grows exponentially (once greater than 40) as the number of customer nodes increases linearly while the DTRC plot does not show an obvious correlation.

When compared with the optimal CVRP solutions, both LNS and DTRC heuristics provide lower objective values by 13.25% and 2.09% respectively (Negative GAP on the table means better objective). Considering a relatively moderate GAP difference between the LNS and optimal CVRP solutions on different problem sets, LNS is quite effective and can be reliable for solving 2EVRPD with the appropriate parameter tuning.

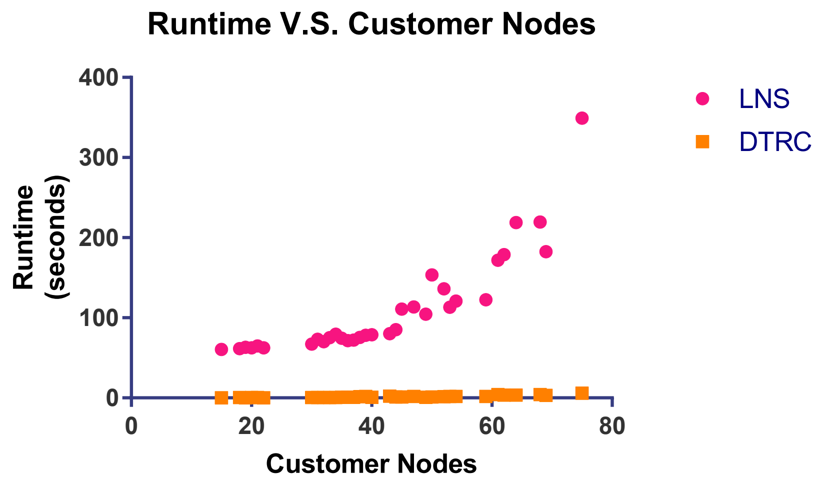


Figure 4.7 The line scatter plot showing the runtime of different problem size

Table 4.3 Results of the proposed heuristics and optimal CVRP on the various instances

							Algorithm Method							
Instance	n	K	Q	Kd	Qd	CVRP (Optimal)	Large Neighborhood search (LNS)				Drone Truck Route Construction (DTRC)			
							GAP (From Truck)	Best	Average	Time	GAP (From Truck)	Best	Average	Time
A-n32-k5	31	5	100	2	35	784	**-13.9**	675	701.8	73.17	**2.04**	800	812.9	0.61
A-n33-k5	32	5	100	2	35	661	**-17.25**	547	566.5	67.93	**-0.76**	656	666.1	0.67
A-n33-k6	32	6	100	2	35	742	**-17.25**	614	633.6	72.7	**-7.55**	686	697.2	0.46
A-n34-k5	33	5	100	2	35	778	**-17.48**	642	653.3	71.18	**-5.4**	736	757.5	0.65
A-n36-k5	35	5	100	2	35	799	**-16.52**	667	683.6	74.37	**-7.51**	739	755.2	0.78
A-n37-k5	36	6	100	2	35	669	**-25.56**	498	543.5	74.63	**-5.08**	635	653.6	0.99
A-n37-k6	36	6	100	2	35	949	**-12.01**	835	850.8	68.23	**-8.43**	869	918.7	0.78
A-n38-k5	37	5	100	2	35	730	**-21.1**	576	622.6	67.42	**0.68**	735	741.3	0.84
A-n39-k5	38	5	100	2	35	822	**-17.76**	676	718.4	71.38	**1.82**	837	865.2	2.07
A-n39-k6	38	6	100	2	35	831	**-16.25**	696	720.9	76.1	**-1.32**	820	833	1.18
A-n44-k6	43	6	100	2	35	937	**-17.4**	774	840.3	80.09	**-4.7**	893	926.6	2.01
A-n45-k6	44	6	100	2	35	944	**-12.5**	826	856.2	85.15	**-1.59**	929	931.3	0.94
A-n46-k7	45	7	100	2	35	914	**-15.75**	770	804.8	110.78	**0.66**	920	921.2	1.32
A-n48-k7	47	7	100	2	35	1073	**-11.28**	952	975	113.57	**-2.52**	1046	1057.5	1.71
A-n53-k7	52	7	100	2	35	1010	**-9.6**	913	949.6	136.06	**-1.09**	999	1039.2	1.44
A-n54-k7	53	7	100	2	35	1167	**-10.45**	1045	1072	113.12	**-4.54**	1114	1127.2	1.83
A-n55-k9	54	9	100	2	35	1073	**-8.2**	985	1011.8	112.09	**-2.89**	1042	1054.6	2.12
A-n62-k8	61	8	100	2	35	1288	**-6.6**	1203	1253.6	171.85	**-2.48**	1256	1273.8	4.3
A-n63-k10	62	10	100	2	35	1314	**-6.7**	1226	1268.8	178.93	**-1.9**	1289	1311.8	3.61
A-n65-k9	64	9	100	2	35	1174	**-10.22**	1054	1106.6	242.33	**-0.85**	1164	1182.1	4.19
A-n69-k9	68	9	100	2	35	1159	**-10.18**	1041	1077.2	219.58	**-4.4**	1108	1122.4	4.15

Table 4.3 continued

Instance	n	K	Q	Kd	Qd	CVRP (Optimal)	Large Neighborhood search (LNS)				Drone Truck Route Construction (DTRC)			
							GAP (From Truck)	Best	Average	Time	GAP (From Truck)	Best	Average	Time
B-n31-k5	30	5	100	2	35	672	**-3.42**	649	653.9	67.33	**-0.74**	667	670.7	0.44
B-n34-k5	33	5	100	2	35	788	**-5.84**	742	751.5	78.9	**-2.16**	771	787.1	0.62
B-n35-k5	34	5	100	2	35	955	**-7.33**	885	888.6	79.49	**-0.84**	947	952.3	0.58
B-n38-k6	37	6	100	2	35	805	**-12.17**	707	730.8	77.2	**-1.12**	796	800.9	0.97
B-n39-k5	38	5	100	2	35	549	**-10.02**	494	503.2	79.39	**-1.46**	541	546.3	0.99
B-n41-k6	40	6	100	2	35	829	**-8.44**	759	823	78.68	**3.98**	862	867.9	0.73
E-n51-k5	50	5	160	2	50	521	**-18.04**	427	449.4	153.52	**-7.29**	483	498.9	1.26
E-n76-k7	75	7	220	2	55	682	**-9.09**	620	633.6	298.06	**-7.92**	628	641.8	8.65
E-n76-k8	75	8	180	2	45	735	**-9.93**	662	698.1	492.36	**-3.4**	710	718.2	7.2
E-n76-k10	75	10	140	2	40	830	**-6.27**	778	805.8	278.3	**-3.98**	797	831.9	4.86
E-n76-k14	75	14	100	2	35	1021	**-3.62**	984	1001.7	327.59	**-0.29**	1018	1032.7	2.61
P-n16-k8	15	8	35	2	20	450	**-1.78**	442	442	60.45	**3.78**	467	467	0.11
P-n19-k2	18	2	160	2	40	212	**-30.66**	147	154.5	61.61	**4.72**	222	228.8	0.35
P-n20-k2	19	2	160	2	40	216	**-29.17**	153	157.9	63.25	**-1.39**	213	215.8	0.32
P-n21-k2	20	2	160	2	40	211	**-32.23**	143	147.5	62.44	**-0.47**	210	231.1	0.52
P-n22-k2	21	2	160	2	40	216	**-31.48**	148	156.9	64.89	**0**	216	216	0.52
P-n23-k8	22	8	40	2	20	529	**-4.35**	506	506.6	62.63	**-0.95**	524	524	0.18
P-n40-k5	39	5	140	2	40	458	**-21.83**	358	374.6	78.25	**-1.31**	452	477.3	1.9
P-n45-k5	44	5	150	2	40	510	**-19.22**	412	422.9	85.42	**-5.69**	481	497.3	1.58
P-n50-k7	49	7	150	2	40	554	**-17.15**	459	500.7	106.63	**-0.36**	552	557.6	0.52
P-n50-k8	49	8	120	2	40	631	**-10.3**	566	577.7	92.69	**-6.66**	589	596.2	1.11
P-n50-k10	49	10	100	2	35	696	**-12.79**	607	664.8	114.42	**0.29**	698	705.6	1.28
P-n55-k7	54	7	170	2	45	568	**-13.91**	489	534.3	143.75	**-3.7**	547	565.5	2.65
P-n55-k10	54	10	115	2	40	694	**-9.65**	627	660.9	138.96	**-1.59**	683	690.5	1.8
P-n55-k15	54	15	70	2	38	989	**-9.71**	893	925.9	88.79	**-5.76**	932	936.6	1.21
P-n60-k10	59	10	120	2	40	744	**-5.91**	700	720.1	125.95	**-2.02**	729	735.5	2.16
P-n60-k15	59	15	80	2	30	968	**-5.37**	916	953	118.78	**0.83**	976	989.7	1.5
P-n65-k10	64	10	130	2	40	792	**-10.73**	707	748.7	195.61	**-1.14**	783	799.6	2.82
P-n70-k10	69	10	135	2	40	827	**-8.22**	759	809.5	182.6	**0**	827	839.3	3.06
						Average	**-13.25**			122.77	**-2.09**			1.78

4.5.4 Sensitivity Analysis

Sensitivity analyses were conducted to evaluate the impact of major parameters and components of the method. We are specifically interested in making a comparison between a different number of drones and see how it would affect the performance of the 2EVRPD solution. In addition, we want to see the impact of our new multiple drops feature when comparing with the typical single drop proposed by Murray and Chu (2015). Four different operational settings are evaluated including 1.) Single drone / Single drop 2.) Single drone / Multiple drops 3.) Two drones / Single drop and 4.) Two drones / Multiple drops. All four types of operations are compared with the optimal solutions from the classical truck only delivery (CVRP).

Using the same benchmark instances as Section 4.5.3, we conducted the experiment using our proposed heuristics for four types of operations. Each instance was run 10 times and we recorded all the experimental results in Table 4.4. We use these results to generate the statistical plot in Figure 4.8 showing the percentage GAP between each operation objective and CVRP objective for all instances with the mean and standard deviation. Negative GAP indicates an improvement over the truck alone operation

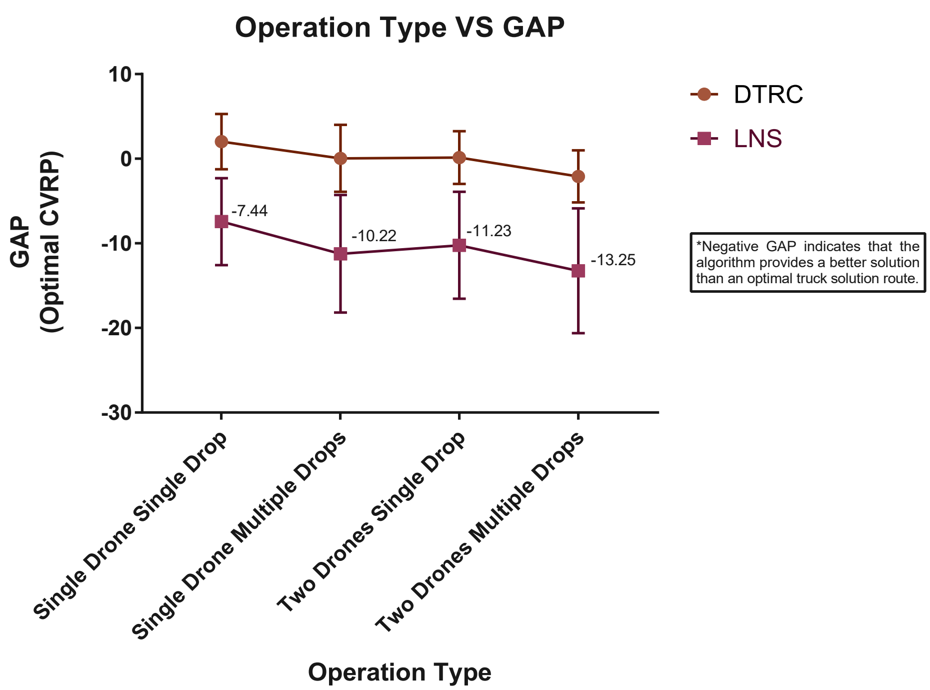


Figure 4.8 The comparison of different operational types using the GAP from a truck alone delivery

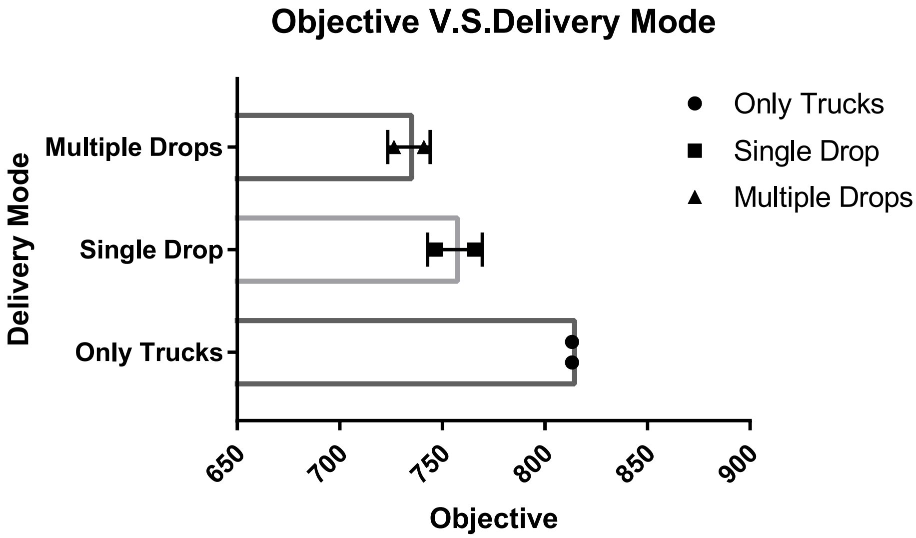


Figure 4.9 The comparison of objective values among different delivery modes

In general, all four types of operations' solutions solved by the LNS algorithm provide the GAP from an optimal CVRP solution much lower than the ones solved by the DTRC algorithm. Among the types of operations, Two drones / Multiple drops operation provides the best solution (with the mean GAP of -13.25%) and Single drone / Single drop provides the worst (with the mean GAP of -7.44%). As we expect, more number of drones leads to lower total routes delivery time. In addition, we report the solution objectives among different types of delivery features as shown in Figure 4.9 with the lowest mean objective value from using trucks and drones with multiple drops feature followed by using trucks and drones with a single drop feature, and only trucks accordingly. It can also be observed that Single drone / Multiple drops operation provides a slightly better solution (with the mean GAP of -11.23%) than Two Drones / Single drops operation (with the mean GAP of -10.22%). The results demonstrate the potential benefit of implementing trucks and drones together with the multiple drops feature in order to significantly reduce the total travel time of the entire delivery.

Table 4.4 Results of the proposed heuristics and optimal CVRP in different types of operations

Instance	n	K	Q	Qd	CVRP (Optimal)	Single Drone												Two Drones											
						Single Drop						Multiple Drops						Single Drop						Multiple Drops					
						DTRC			LNS			DTRC			LNS			DTRC			LNS			DTRC			LNS		
						Best	Average	Time	Best	Average	Time	Best	Average	Time	Best	Average	Time	Best	Average	Time	Best	Average	Time	Best	Average	Time	Best	Average	Time
A-n32-k5	31	5	100	35	784	855.00	857.40	0.46	713.00	731.80	72.12	795.00	799.50	0.53	692.00	719.00	65.59	820.00	833.60	0.49	713.00	731.80	72.12	800.00	812.90	0.61	675.00	701.80	73.17
A-n33-k5	32	5	100	35	661	696.00	711.80	0.59	591.00	597.30	68.84	676.00	689.50	0.58	548.00	566.30	68.72	687.00	701.50	0.60	559.00	571.70	77.17	656.00	666.10	0.67	547.00	566.50	67.93
A-n33-k6	32	6	100	35	742	717.00	722.00	0.45	635.00	650.00	64.83	697.00	697.00	0.45	627.00	635.60	64.05	717.00	724.00	0.46	623.00	633.00	70.41	686.00	697.20	0.46	614.00	633.60	72.70
A-n34-k5	33	5	100	35	778	766.00	770.20	0.64	670.00	677.60	66.23	766.00	770.20	0.63	634.00	647.50	70.05	766.00	770.20	0.64	652.00	669.10	74.41	736.00	757.50	0.65	642.00	653.30	71.18
A-n36-k5	35	5	100	35	799	835.00	839.00	0.77	694.00	725.30	64.77	751.00	751.00	0.76	659.00	682.20	69.05	785.00	790.60	0.78	683.00	699.90	71.96	739.00	755.20	0.78	667.00	683.60	74.37
A-n37-k5	36	6	100	35	669	677.00	678.20	0.99	564.00	594.70	72.00	659.00	663.80	1.00	500.00	538.00	70.11	650.00	655.00	1.01	551.00	584.30	75.70	635.00	653.60	0.99	498.00	543.50	74.63
A-n37-k6	36	6	100	35	949	904.00	936.10	0.77	852.00	876.50	65.80	909.00	909.00	0.77	828.00	846.00	67.79	915.00	934.10	0.79	842.00	869.70	66.93	869.00	918.70	0.78	835.00	850.80	68.23
A-n38-k5	37	5	100	35	730	765.00	766.30	0.83	651.00	676.00	75.25	717.00	717.00	0.83	601.00	635.00	70.47	763.00	765.30	0.85	618.00	638.00	71.80	735.00	741.30	0.84	576.00	622.60	67.42
A-n39-k5	38	5	100	35	822	883.00	890.20	2.05	749.00	777.00	67.90	863.00	881.00	2.06	717.00	736.50	72.85	862.00	882.70	2.07	718.00	745.90	70.71	837.00	865.20	2.07	676.00	718.40	71.38
A-n39-k6	38	6	100	35	831	826.00	830.50	1.16	755.00	766.70	71.72	850.00	850.00	1.16	660.00	723.60	80.83	799.00	806.20	1.18	710.00	738.10	76.56	820.00	833.00	1.18	696.00	720.90	76.10
A-n44-k6	43	6	100	35	937	939.00	956.10	2.05	886.00	908.60	67.41	902.00	903.70	1.99	835.00	860.00	73.16	917.00	937.50	2.01	839.00	880.10	78.03	893.00	926.60	2.01	774.00	840.30	80.09
A-n45-k6	44	6	100	35	944	971.00	973.00	0.99	890.00	906.50	81.65	941.00	941.00	0.91	849.00	872.10	76.85	966.00	970.00	0.94	868.00	891.70	86.70	929.00	931.30	0.94	826.00	856.20	85.15
A-n46-k7	45	7	100	35	914	954.00	959.80	1.35	825.00	847.10	78.92	938.00	940.10	1.34	810.00	827.00	86.17	912.00	922.50	1.41	795.00	828.80	91.74	920.00	921.20	1.32	770.00	804.80	110.78
A-n48-k7	47	7	100	35	1073	1092.00	1112.40	1.69	975.00	1017.40	85.79	1071.00	1078.30	1.65	956.00	987.50	86.82	1084.00	1091.80	1.72	966.00	1015.90	90.92	1046.00	1057.50	1.71	952.00	975.00	113.57
A-n53-k7	52	7	100	35	1010	1069.00	1073.10	1.37	969.00	997.10	93.03	1025.00	1027.10	1.37	914.00	953.10	126.34	1053.00	1057.70	1.39	931.00	991.70	124.93	999.00	1039.20	1.44	913.00	949.60	136.06
A-n54-k7	53	7	100	35	1167	1150.00	1160.60	1.83	1084.00	1108.70	84.73	1136.00	1144.20	1.83	1050.00	1075.40	97.71	1117.00	1134.20	1.84	1067.00	1099.50	107.21	1114.00	1127.20	1.83	1045.00	1072.00	113.12
A-n55-k9	54	9	100	35	1073	1086.00	1088.60	2.10	1015.00	1041.80	86.01	1049.00	1049.00	2.09	973.00	1000.90	89.81	1083.00	1085.10	2.18	1011.00	1032.00	109.57	1042.00	1054.60	2.12	985.00	1011.80	112.09
A-n62-k8	61	8	100	35	1288	1315.00	1320.10	4.28	1246.00	1282.30	156.30	1339.00	1340.00	4.27	1244.00	1270.40	95.75	1347.00	1353.00	4.28	1230.00	1261.40	297.44	1256.00	1273.80	4.30	1203.00	1253.60	171.85
A-n63-k10	62	10	100	35	1314	1334.00	1336.00	3.61	1292.00	1317.50	91.54	1299.00	1300.40	3.61	1251.00	1270.10	115.80	1353.00	1357.80	3.60	1235.00	1283.70	225.83	1289.00	1311.80	3.61	1226.00	1268.80	178.93
A-n65-k9	64	9	100	35	1174	1223.00	1231.90	4.01	1148.00	1178.10	150.38	1184.00	1186.90	4.04	1121.00	1140.60	115.00	1208.00	1224.90	4.08	1119.00	1154.10	201.99	1164.00	1182.10	4.19	1054.00	1106.60	242.33
A-n69-k9	68	9	100	35	1159	1166.00	1189.50	4.10	1088.00	1155.90	134.74	1134.00	1138.80	4.12	1075.00	1111.30	105.34	1151.00	1168.40	4.16	1086.00	1130.30	246.72	1108.00	1122.40	4.15	1041.00	1077.20	219.58
B-n31-k5	30	5	100	35	672	672.00	675.20	0.43	657.00	658.80	65.94	669.00	670.20	0.43	648.00	651.50	66.17	675.00	677.00	0.44	655.00	657.50	72.67	667.00	670.70	0.44	649.00	653.90	67.33
B-n34-k5	33	5	100	35	788	784.00	789.40	0.60	751.00	761.60	65.79	772.00	788.00	0.61	734.00	747.90	70.87	787.00	792.50	0.62	749.00	758.60	70.22	771.00	787.10	0.62	742.00	751.50	78.90
B-n35-k5	34	5	100	35	955	965.00	978.40	0.57	904.00	909.70	69.51	969.00	977.30	0.57	887.00	891.10	72.00	966.00	975.60	0.59	892.00	896.30	78.95	947.00	952.30	0.58	885.00	888.60	79.49
B-n38-k6	37	6	100	35	805	835.00	837.90	0.95	740.00	756.30	73.04	809.00	810.60	0.95	708.00	730.20	68.88	835.00	839.30	0.97	725.00	741.60	83.52	796.00	800.90	0.97	707.00	730.80	77.20
B-n39-k5	38	5	100	35	549	543.00	543.90	1.03	513.00	519.50	68.96	551.00	551.30	0.97	492.00	503.50	75.63	541.00	543.80	0.99	505.00	510.80	77.00	541.00	546.30	0.99	494.00	503.20	79.39
B-n41-k6	40	6	100	35	829	885.00	889.10	0.71	822.00	844.80	73.27	873.00	873.00	0.72	817.00	833.10	71.52	886.00	887.90	0.72	813.00	841.40	70.76	862.00	867.90	0.73	759.00	823.00	78.68
E-n51-k5	50	5	160	50	521	571.00	581.10	1.43	466.00	495.10	258.18	499.00	516.90	1.04	446.00	475.20	124.02	522.00	537.60	1.20	456.00	471.10	248.96	483.00	498.90	1.26	427.00	449.40	153.52
E-n76-k7	75	7	220	55	682	712.00	736.00	8.70	664.00	698.80	384.89	630.00	652.30	8.72	616.00	635.20	273.20	651.00	712.30	8.71	609.00	657.20	323.43	628.00	641.80	8.65	620.00	633.60	298.06
E-n76-k8	75	8	180	45	735	745.00	774.80	7.20	696.00	743.30	220.64	729.00	735.60	7.20	698.00	725.80	131.00	701.00	725.30	7.21	673.00	704.00	442.96	710.00	718.20	7.20	662.00	698.11	492.36
E-n76-k10	75	10	140	40	830	875.00	888.40	4.85	831.00	854.70	237.54	858.00	858.00	4.86	805.00	830.70	235.22	838.00	855.10	4.94	808.00	827.30	282.09	797.00	831.90	4.86	778.00	805.80	278.30
E-n76-k14	75	14	100	35	1021	1049.00	1063.70	2.56	1014.00	1040.40	160.61	1038.00	1038.40	2.58	1007.00	1018.70	125.25	1024.00	1029.30	2.62	997.00	1017.50	220.78	1018.00	1032.70	2.61	984.00	1001.70	327.59
P-n16-k8	15	8	35	20	450	469.00	469.00	0.10	444.00	444.00	60.35	467.00	467.00	0.10	442.00	442.00	60.31	469.00	469.00	0.11	444.00	444.00	60.38	467.00	467.00	0.11	442.00	442.00	60.45
P-n19-k2	18	2	160	40	212	221.00	223.40	0.34	175.00	175.80	61.16	246.00	246.00	0.33	164.00	165.90	60.99	220.00	220.00	0.34	159.00	164.20	61.54	222.00	228.80	0.35	147.00	154.50	61.61
P-n20-k2	19	2	160	40	216	215.00	240.50	0.30	177.00	177.30	61.29	234.00	234.00	0.31	165.00	167.30	61.38	203.00	217.30	0.32	158.00	167.80	62.10	213.00	215.80	0.32	153.00	157.90	63.25
P-n21-k2	20	2	160	40	211	205.00	209.40	0.51	170.00	174.40	61.75	203.00	208.40	0.51	159.00	161.20	62.36	199.00	202.00	0.51	159.00	163.10	62.95	210.00	231.10	0.52	143.00	147.50	62.44
P-n22-k2	21	2	160	40	216	227.00	237.40	0.51	177.00	180.50	61.52	217.00	227.80	0.52	156.00	164.40	62.34	221.00	221.00	0.53	166.00	168.70	63.13	216.00	216.00	0.52	148.00	156.90	64.89
P-n23-k8	22	8	40	20	529	527.00	527.00	0.17	500.00	505.40	60.98	524.00	524.00	0.17	500.00	505.20	61.09	527.00	527.00	0.18	506.00	506.20	60.89	524.00	524.00	0.18	506.00	506.60	62.63
P-n40-k5	39	5	140	40	458	462.00	464.80	1.90	399.00	406.20	71.69	472.00	472.00	1.88	358.00	381.50	70.83	457.00	461.40	1.92	377.00	394.10	90.14	452.00	477.30	1.90	358.00	374.60	78.25
P-n45-k5	44	5	150	40	510	527.00	541.30	1.57	448.00	464.50	81.62	493.00	493.00	1.57	400.00	424.20	73.34	504.00	527.90	1.58	411.00	440.20	107.91	481.00	497.30	1.58	412.00	422.90	85.42
P-n50-k7	49	7	150	40	554	576.00	584.30	0.35	522.00	542.00	89.50	575.00	575.50	0.35	492.00	516.70	93.39	552.00	567.20	0.46	501.00	520.50	92.24	552.00	557.60	0.52	459.00	500.70	106.63
P-n50-k8	49	8	120	40	631	606.00	616.30	1.08	581.00	592.60	83.48	607.00	622.20	1.09	557.00	578.40	82.59	603.00	609.40	1.10	576.00	586.70	92.64	589.00	596.20	1.11	566.00	577.70	92.69
P-n50-k10	49	10	100	35	696	702.00	718.90	1.27	669.00	680.40	81.66	699.00	703.50	1.27	651.00	668.20	74.07	704.00	714.90	1.28	660.00	674.10	84.93	698.00	705.60	1.28	607.00	664.80	114.42
P-n55-k7	54	7	170	45	568	581.00	611.90	2.65	517.00	541.30	131.02	569.00	589.80	2.73	496.00	524.50	105.40	573.00	596.20	2.67	498.00	521.50	195.16	547.00	565.50	2.65	489.00	534.30	143.75
P-n55-k10	54	10	115	40	694	696.00	700.60	1.77	656.00	679.60	88.88	693.00	697.60	1.79	640.00	666.30	96.45	697.00	703.60	1.81	657.00	676.30	108.96	683.00	690.50	1.80	627.00	660.90	138.96
P-n55-k15	54	15	70	38	989	955.00	960.00	1.20	928.00	945.10	81.30	932.00	932.50	1.18	917.00	928.20	77.63	950.00	956.60	1.15	914.00	945.80	80.74	932.00	936.60	1.21	893.00	925.90	88.79
P-n60-k10	59	10	120	40	744	738.00	750.40	2.14	723.00	738.30	101.36	737.00	747.70	2.11	705.00	731.10	89.17	736.00	749.90	2.17	689.00	730.10	146.77	729.00	735.50	2.16	700.00	720.10	125.95
P-n60-k15	59	15	80	30	968	966.00	978.80	1.48	940.00	956.20	82.37	972.00	982.90	1.47	937.00	959.20	73.69	966.00	978.80	1.49	938.00	958.80	92.92	976.00	989.70	1.50	916.00	953.00	118.78
P-n65-k10	64	10	130	40	792	838.00	863.40	2.89	782.00	809.50	154.63	810.00	817.80	2.80	761.00	780.10	121.89	794.00	815.90	2.81	741.00	780.00	192.06	783.00	799.60	2.82	707.00	748.70	195.61
P-n70-k10	69	10	135	40	827	854.00	881.70	2.96	833.00	868.10	121.02	812.00	828.00	3.01	774.00	816.40	103.01	838.00	856.00	2.98	787.00	823.60	240.26	827.00	839.30	3.06	759.00	809.50	182.60

4.6 Conclusion

In this chapter, we propose a new routing model, the Two Echelon Vehicle Routing Problems with Drones (2EVRPD), which implements both trucks and drones in the last-mile delivery. The model is a variation of the classic CVRP problem and the extension of the previous FSTSP model. We generalize our previous work mTSPD, which we allow multiple drones and multiple trucks to perform deliveries, to 2EVRPD, which includes capacity constraints and multiple drops feature. The MIP formulation is mathematically constructed to model the 2EVRPD to solve for an optimal solution for the small-size instances, in which the results are shown in Section 4.5.1. To solve the large-size instances, we develop two heuristics: 1.) Drone Truck Route Construction (DTRC), which is a constructive heuristic used for creating an initial 2EVRPD solution from an empty route, and 2.) LNS, which iteratively improves the 2EVRPD solution using destroy and repair principles. Using these heuristics, we can solve and obtain better solutions than CPLEX solver under the same computational time as shown in Section 4.5.2. LNS heuristic generates higher quality than DTRC when tested on the various CVRP benchmark problems. LNS also returns the solutions better than the reported optimal CVRP on the same instances (Section 4.5.3), which implies that using this new approach provides shorter delivery time than simply using trucks alone in the operation. Lastly, the result from the sensitivity analysis in Section 4.5.4 shows that implementing a multiple drops feature improves the solution quality from the typical single drop feature.

CHAPTER 5. INTEGRATED VEHICLE ROUTING PROBLEM WITH DRONES

5.1 Problem Description

In this chapter, we consider another hybrid truck drone routing model which extends the FSTSP by allowing multiple drones and multiple trucks to make deliveries with the consideration of both trucks' and drones' capacities. We refer to this specific type of truck with the drone equipped on top as a "hybrid truck." In addition to that, we include two more types of vehicle fleets in the routing operation including a "large drone" or "cargo drone" and a "traditional truck". A large drone is a new generation of plane-sized autonomous delivery vehicles, which is capable of carrying hundreds of pounds for hundreds of miles which is significantly much larger in size and longer in flying duration (Hawkins, 2019). Large drones offer benefits of speed similar to the small drones with more endurance and capacity (Shivakumar, 2019). However, it comes with a heavy cost and has been recently tested in only a specific region. Unlike a small drone that is designed to carry a single item one at a time, a large drone can carry multiple packages and make multiple deliveries to different customers before merging with the truck. Lastly, a traditional truck is simply a truck without a small drone and is used in the current last-mile delivery. Figure 5.1a.) and 5.1b.) on the next page illustrate the hybrid truck with a small drone and the large drone we discussed.

To the best of our knowledge, none of the previous studies has integrated and combined different types of vehicle fleets that involve drones/large drones together. We intend to study and investigate the benefits of this approach in comparison with the other existing drone routing models that are used for the last-mile delivery. We call this model the "Integrated Vehicle Routing Problem with Drones" (I-VRPD). Figure 5.2 represents a simple I-VRPD and its solution in which three different types of vehicles are used in the setting. As illustrated, a hybrid truck contains two drones and each drone can fly to serve one customer before immediately returning to the truck. The grey solid line represents the solution for the traditional truck and the dashed line represents the solution from a large drone.

a.)

b.)

Figure 5.1 Examples of a.) a hybrid truck with small drone and b.) a large/cargo drone for delivery purpose (Hawkins, 2019)

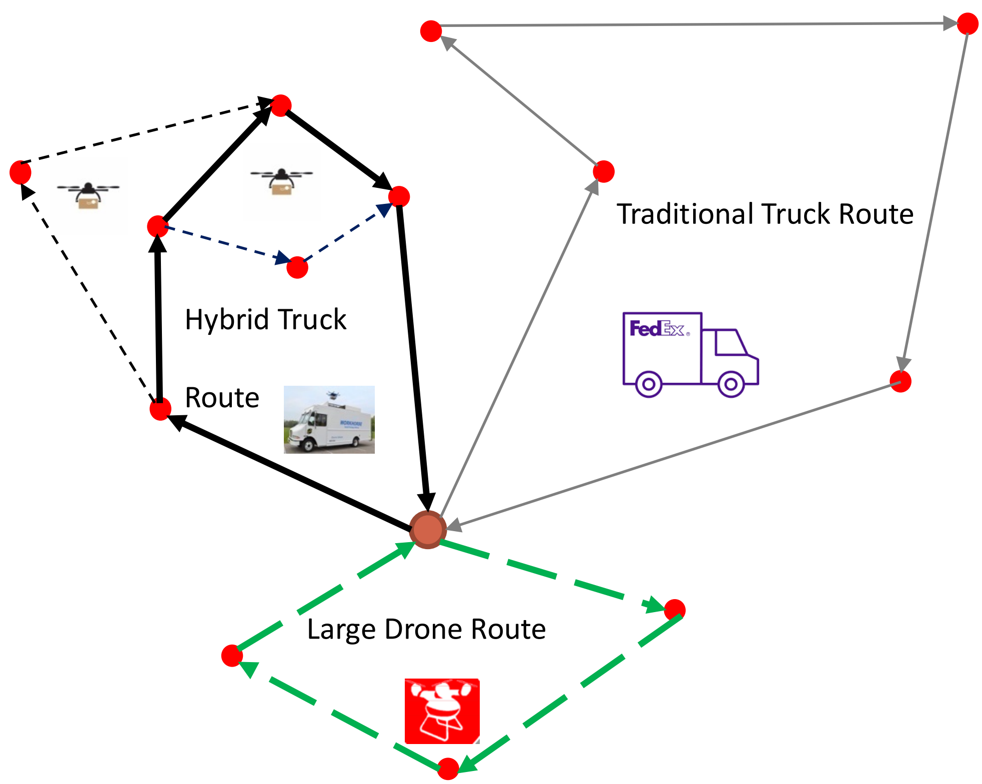


Figure 5.2 Illustration of the Integrated Vehicle Routing Problems with Drones (I-VRPD)

It is important to note that each vehicle fleet is operated independently and all vehicle units in each fleet are assumed to be homogenous. Each hybrid truck has limited space to carry only a specific number of small drones. A small drone can carry a single package one at the time while a large drone can carry multiple packages per trip. Both hybrid trucks and traditional trucks have the same capacity and unlimited endurance. A small drone and a large drone each uniquely have a limited amount of battery capacity, which determines how long the drones can travel before receiving a battery swap or a battery recharge.

5.2 Assumptions and Contributions

5.2.1 Assumptions

For the hybrid truck operation, we assume the following assumptions:

- Multiple drones are not allowed to be launched or retrieved at the same node at any given time.
- Small drones can only merge with a hybrid truck at a customer node and are not allowed to merge with a hybrid truck in any intermediate location.
- Besides, the time of both hybrid truck and small drone at the customer locations must be adjusted to be the same. In other words, they must wait for each other whenever one arrives at the customer node before the other.
- The set-up and recovery times when the small drone is launched or retrieved at a particular node can be negligible since their values are small compared to the hybrid truck and small drone travel time.
- And finally, the small drone must go back to the hybrid truck from which it was launched (it cannot be merged with different hybrid trucks) and it is important to keep track of when the small drones are available for the launch.

The proposed model aims at finding a combination of solution routes from different fleets of vehicles such that the demands of all customers are satisfied while the total delivery time is minimized. We believe that the successful integration of drones combining with other vehicle types could bring about cost efficiency and reduce the total delivery time of the last-mile delivery.

5.2.2 Contributions

We propose a new MIP model and heuristic algorithms to solve a new problem, the Integrated Vehicle Routing Problems with Drones (I-VRPD). The main contributions of this paper are the following:

1. We introduce a Mixed Integer Program (MIP) formulation for the I-VRPD. The model might be solved by any standard MILP solver, e.g., GAM and IBM CPLEX. They can handle small-size problems.
2. We propose two metaheuristics to solve I-VRPD based on Variable Neighborhood Search (VNS) and Large Neighborhood Search (LNS). In particular, an initial I-VRPD is generated based on the classical VRP solution and will be iteratively improved by the exploitation of local improvement procedures and exploration procedures. Similarly, the LNS will repeatedly search for a better solution using four destroy operators and four repair operators.

3. We conduct a case study and numerical experiments on different types of problems using both MIP and heuristics approaches. The case study and small-size problems can be solved directly by the MIP solver and the medium/large-size problems can be solved by the heuristics. The results are compared with the classical VRP optimal solutions and other VRP with drone (VRPD) routing models on the CVRP benchmark problems.

The remainder of this chapter is structured as follows. Section 5.3 presents the formal definition of the I-VRPD model and its mathematical MIP formulation. Section 5.4 presents the two heuristics approaches: Variable Neighborhood Search and Large Neighborhood Search, to solve the proposed model along with the pseudocodes in detail. In Section 5.5, we provide the results of the case study and the numerical experiments on the different test instances and benchmark problems with the analysis of the performance of the algorithm. Section 5.6 concludes the paper and provides discussions for future research.

5.3 Mathematical Formulation

The I-VRPD is a combinatorial optimization problem that can be formulated by Mixed Integer Programming (MIP). It is defined on a directed graph G = (V, E), where V is the set of n nodes representing customers with one depot and E is the set of arcs. The new integrated system provides flexibility and options for clients to receive items from any of the vehicle fleets in which each type of fleet has its advantage. While a traditional truck delivers packages with large-volume or huge load to customers who might be located far from the depot, the hybrid truck with small drones can deliver items with small-volume or light load to the customers who are located close to the depot. Additionally, a large drone can carry multiple heavy items with longer battery capacity than the small drones which is ideally a great fit for customers who want faster delivery for their large items. The resulting configuration is expected to reduce the total system operation cost, improve overall delivery speed, and increase long term customer satisfaction. The following notations describe data sets, model parameters, and decision variables used to formulate the I-VRPD.

Indices

i, j, p	Represent customers, and depot index
k	Represent hybrid truck index
kd	Represent small drone index
vt	Represent traditional truck index
vd	Represent large drone index

Sets

C	Set of customers, $\{1,\ 2,\ 3,\ 4,\ 5,\ 6, \dots, n\}$
C_0	Set of customer nodes including the starting depot, $C \cup \{0(s)\}$
C_+	Set of customer nodes including the ending depot, $C \cup \{0(r)\}$
D	Set of demands for all customers, $\{d_1, d_2, d_3, \dots, d_n\}$
K	Set of hybrid trucks, $\{1,2,3,\dots,k\}$
KD	Set of small drones, $\{1,2,3,\dots,kd\}$
VT	Set of traditional trucks, $\{1,2,3,\dots,vt\}$
VD	Set of large drones, $\{1,2,3,\dots,vd\}$

Parameters

$\tau_{i,j}^{T}$	Truck travel time between nodes *i* and *j*
$\tau_{i,j}^{D}$	Drone travel time between nodes *i* and *j*
Q	Hybrid-trucks capacity (Same for all hybrid trucks)
QD	Small Drone capacity (Same for all drones)
QVT	Traditional truck capacity (Same for all traditional trucks)
QVD	Large drone capacity (Same for all large drones)
B	Battery limit for drones (Small drone's battery life)
BVD	Battery limit for large drones (Large drone's battery life)
D_i	Customer demand at each node *i*

Variables

$x_{i,j}^{k}$	1 if a hybrid truck *k* traverses arc (i, j) from node *i* to node *j*; otherwise, 0
$y_{i,j,p}^{kd,k}$	1 if a small drone *kd* of truck *k* traverses arc (i, j, p) from node *i* to node *j* and return to node *p*; otherwise, 0
$xvt_{i,j}^{vt}$	1 if a traditional truck *vt* travels from node *i* to node *j*; otherwise, 0
$xvd_{i,j}^{vd}$	1 if a large drone *vd* travels from node *i* to node *j*; otherwise, 0
$yt_i^k, yvt_i^{vt}, yvd_i^{vd}, yd_i^{kd,k}$	indicates whether a hybrid truck, a traditional truck, a large drone and a small drone serve customer node *i* accordingly or not
tt_j^k	Hybrid truck *k* arrival time at node *j*
$dt_j^{kd,k}$	Small drone *kd* of hybrid truck *k* arrival time at node *j*
tvt_i^{vt}	Traditional truck *vt* arrival time at node *j*
tvd_i^{vd}	Large drone *vd* arrival time at node *j*
bc_i^{vd}	The battery consumption of a large drone *vd* at node *i*
$la_i^{kd,k}$	The state of node *i* which can launch a small drone *kd* of hybrid truck *k* (0 if launchable state, 1 unlaunchable state)
$u_i^k, u_i^{vt}, u_i^{vd}$	Auxiliary variable for VRP subtour elimination constraints for all vehicle fleets

Three vehicle fleet types are defined as a set of *K = {1, 2, 3, 4,…,k}, VT = {1,2,3,…,vt}* and *VD = {1,2,3,…,vd}* which represent a hybrid truck fleet, a traditional truck fleet and a large drone fleet accordingly. They must carry the amount of load less than their capacities (*Q* for a hybrid truck, *QVT* for a traditional truck and *QVD* for a large truck). Each fleet type consists of a certain number of homogeneous vehicle units. Each unit of a hybrid truck is attached with a set of small drones, *KD = {1, 2, 3, 4,…,kd},* each can handle load up to *QD.* The amount of load is measured by the weight unit for all vehicles. In addition, the drone's travel capability is restricted by its battery limitation which is defined as *B* for a small drone and *BVD* for a large drone. Each customer *i* (*i* = 1,2,3,…, *n*) is associated with a known nonnegative demand, D_i, to be delivered, and the depot has a fictitious demand $D_o = 0$.

Let $\tau^T_{i,j}$ be a truck travel time associated with *E,* $(i,j) \in E$ and $\tau^D_{i,j}$ be a drone travel time associated with *E,* $(i,j) \in E$, differentiating the travel times for the truck and drone accounts for each vehicle's unique travel speed. The I-VRPD is said to be symmetric if $\tau^T_{i,j} = \tau^T_{j,i}$ and $\tau^D_{i,j} = \tau^D_{j,i}$ and asymmetric otherwise. Please note that the travel times for truck and drone matrix satisfy the triangle inequality $\tau^T_{i,k}+\tau^T_{k,j} \geq \tau^T_{i,j}$.

For readability purpose, we denote the set of customer nodes by $C = \{1, 2, 3, 4, 5, 6, \dots, n\}$ and additionally define a set $C_0 = C \cup \{0(s)\}$ as the set of customer nodes including the starting depot, and set $C_+ = C \cup \{0(r)\}$ as the set of customer nodes including the ending depot. We define the following decision variables: Let $x^k_{i,j}$ be equal to 1 if a hybrid truck k travels along the arc *(i, j)* ∈ *E* and 0 otherwise. This refers to the situation when the truck travels from node *i* ∈ C_0 to *j* ∈ C_+ where *i≠j.* Similarly, let $xvt^{vt}_{i,j}$ and $xvd^{vd}_{i,j}$ be equal to 1 if a traditional truck *vt* and a large drone *vd* travel along the arc *(i, j)* ∈ *E* and 0 otherwise. Let $y^{kd,k}_{i,j,p}$ be equal to 1 if a small drone *kd* of hybrid truck *k* travels along the arc *(i, j)* and *(j, p)* ∈ *E* and 0 otherwise. Additionally, we use variables yt^k_i, yvt^{vt}_i, yvd^{vd}_i and $yd^{kd,k}_i$ to indicate whether a hybrid truck, a traditional truck, a large drone and a small drone serve customer node *i* accordingly or not

For tracking operational time, we denote the variable tt^k_j as the hybrid truck *k* arrival time at node $j \in C_+$ and $dt^{kd,k}_j$ as the small drone *kd* of hybrid truck *k* arrival time at node $j \in C_+$. tt^k_j and $dt^{kd,k}_j$ are are adjusted to be the same in any node j. The variable tvt^{vt}_j and tvt^{vd}_j represent the traditional truck *vt* and large drone arrival time at node $j \in C_+$ accordingly. Lastly, we define other the auxiliary decision variables including 1) u^k_i u^{vt}_i, u^{vd}_i which are used in the VRP subtour elimination constraints (Desrochers & Laporte, 1991), 2) $la^{kd,k}_i$ which is used to indicate the status of whether a small drone *kd* of hybrid truck *k* can be launched from node *i* or not.

The proposed MIP formulation of I-VRPD is presented as follows:

Objective

$$\text{minimize} \sum_{k\in K} tt_{0(r)}^{k} + \sum_{vt\in VT} tvt_{0(r)}^{vt} + \sum_{vd\in VD} tvd_{0(r)}^{vd} \tag{1}$$

The objective function (1) minimizes the total arrival time of all vehicle units across different fleets at the depot.

Subject to

$$\sum_{k\in K}\sum_{kd\in KD} yd_i^{kd,k} + \sum_{k\in K} yt_i^{k} + \sum_{vt\in VT} yvt_i^{vt} + \sum_{vd\in VD} yvd_i^{vd} = 1;\ \forall i \in C \tag{2}$$

Constraints (2) ensure that each customer will receive the package from one of the following vehicles: a hybrid truck, a small drone, a traditional truck and a large drone exactly once.

$$\sum_{i\in C_+} x_{0(s),i}^{k} = 1;\ \forall k \in K \tag{3}$$

$$\sum_{i\in C_0} x_{i,0(r)}^{k} = 1;\ \forall k \in K \tag{4}$$

$$\sum_{j\in C_+} x_{i,j}^{k} = \sum_{j\in C_0} x_{j,i}^{k} = yt_i^{k};\ \forall i \in C,\ \ \forall k \in K \tag{5}$$

Constraints (3) and (4) impose that each hybrid truck must depart from and arrive at the depot. Constraints (5) ensure the flow conservation of the hybrid truck route at each node *i,* which guarantees that whenever the hybrid truck *k* arrives at a node, it must depart from the node as well.

$$\sum_{i\in C_+} xvt_{0(s),i}^{vt} = 1;\ \forall vt \in VT \tag{6}$$

$$\sum_{i \in C_0} xvt_{i,0(r),}^{vt} = 1;\ \forall vt \in VT \tag{7}$$

$$\sum_{j \in C_+} xvt_{i,j}^{vt} = \sum_{j \in C_0} xvt_{j,i}^{vt} = yvt_i^{vt};\ \forall i \in C,\ \ \forall vt \in VT \tag{8}$$

Similarly, the sets of constraints (6) to (8) and (9) to (11) impose the same restriction for a traditional truck and a large drone which basically ensures the flow conservation and guarantee the departure and the arrival to the depot.

$$\sum_{i \in C_+} xvd_{0(s),i}^{vd} = 1;\ \forall vd \in VD \tag{9}$$

$$\sum_{i \in C_0} xvt_{i,0(r),}^{vt} = 1;\ \forall vt \in VT \tag{10}$$

$$\sum_{j \in C_+} xvt_{i,j}^{vt} = \sum_{j \in C_0} xvt_{j,i}^{vt} = yvt_i^{vt};\ \forall i \in C,\ \ \forall vt \in VT \tag{11}$$

$$\sum_{i \in C} \sum_{p \in C} y_{i,j,p}^{kd,k} = yd_j^{kd,k};\ \forall j \in C, \forall k \in K, \forall kd \in KD \tag{12}$$

Constraints (12) impose that a customer at node *j* must be served by a small drone when it travels from node *i* to node *j*, and node *p* in order.

$$\sum_{kd \in KD} \sum_{k \in K} \sum_{j \in C} \sum_{p \in C} y_{i,j,p}^{kd,k} \le 1;\ \forall i \in C \tag{13}$$

$$\sum_{kd \in KD} \sum_{k \in K} \sum_{p \in C} \sum_{j \in C} y_{p,j,i}^{kd,k} \le 1;\ \forall i \in C \tag{14}$$

Constraints (13) and (14) represent that at most one drone can depart from and arrive at a hybrid truck at each stop.

$$2y_{i,j,p}^{kd,k} \le \sum_{\substack{h \in C_0 \\ h \neq i}} x_{h,i}^k + \sum_{\substack{l \in C \\ l \neq p}} x_{l,p}^k;\ \forall i,j \in C,\ \forall p \in C, \forall k \in K, \forall kd \in KD \tag{15}$$

Constraints (15) state that a hybrid truck must visit node *i* and node *p* if the drone is launched from node *i* and is retrieved at node *p*.

$$\sum_{kd\in KD}\sum_{k\in K}\sum_{i\in C}\sum_{p\in C} y_{i,j,p}^{kd,k} \le 1 - \sum_{kd\in KD}\sum_{k\in K}\sum_{a\in C}\sum_{b\in C} y_{j,a,b}^{kd,k};\ \forall j \in C \tag{16}$$

$$\sum_{kd\in KD}\sum_{k\in K}\sum_{i\in C}\sum_{p\in C} y_{i,j,p}^{kd,k} \le 1 - \sum_{kd\in KD}\sum_{k\in K}\sum_{a\in C}\sum_{b\in C} y_{a,b,j}^{kd,k};\ \forall j \in C \tag{17}$$

Constraints (16) describe the cases that if the small drone flies from node *i* to node *j* to node *p*, it can not fly from node *j* to make delivery at other nodes. Similarly, constraint (17) enforces that if a drone small departs from node *i* to serve a customer at node *j* and merges with the hybrid truck at node *p*, no other drones can arrive at the delivery node *j*.

The following sets of constraints (18) to (25) guarantee the correct order of launching and landing operation for a hybrid truck fleet by ensuring that a small drone can only be launched if it has never been launched before or was previously launched, successfully completed its job and returned to receive a service at the hybrid truck

$$la_i^{kd,k}\Big(\sum_{j\in C}\sum_{p\in C} y_{p,j,i}^{kd,k}\Big) = 0; \forall i \in N, \forall k \in K, \forall kd \in KD \tag{18}$$

$$la_i^{kd,k}\Big(\sum_{j\in C}\sum_{p\in C} y_{i,j,p}^{kd,k}\Big) = 0; \forall i \in N,\ \forall k \in K, \forall kd \in KD \tag{19}$$

Constraints (18) and (19) enforces that a small drone is not allowed to be launched or land at node *i* once the auxiliary variable $la_i^{kd,k}$ is equal to 1 and vice versa.

$$la_j^{kd,k} \ge 1 - M\Big(2 - x_{i,j}^k - \sum_{q\in C}\sum_{p\in C} y_{i,q,p}^{kd,k} + la_i^{kd,k} + \sum_{a\in C}\sum_{b\in C} y_{a,b,j}^{kd,k}\Big); \forall i, \forall j \in C, \forall k \in K, \forall kd \in KD \tag{20}$$

$$la_j^{kd,k} \le 1 + M\Big(2 - x_{i,j}^k - \sum_{q\in C}\sum_{p\in C} y_{i,q,p}^{kd,k} + la_i^{kd,k} + \sum_{a\in C}\sum_{b\in C} y_{a,b,j}^{kd,k}\Big); \forall i, \forall j \in C, \forall k \in K, \forall kd \in KD \tag{21}$$

Constraints (20) to (23) ensure that if the small drone is launched from node *i* and has not returned at node *p,* then the auxiliary variable $la_j^{kd,k}$must be equal to 1, the state which no arc drone leaves or enters node *j.* Constraints (20) and (21) deal with the case in which the small drone is launched from node *i,* and the hybrid truck travels from node *i* to node *j* at which the small drone has not yet returned.

$$la_j^{kd,k} \geq 1 - M(2 - x_{i,j}^k - la_i^{kd,k} + \sum_{a\in C}\sum_{b\in C} y_{a,b,j}^{kd,k})\ ; \forall i, \forall j \in C,\ \forall k \in K, \forall kd \in KD \tag{22}$$

$$la_j^{kd,k} \leq 1 + M(2 - x_{i,j}^k - la_i^{kd,k} + \sum_{a\in C}\sum_{b\in C} y_{a,b,j}^{kd,k})\ ; \forall i, \forall j \in C,\ \forall k \in K, \forall kd \in KD \tag{23}$$

Constraints (22) and (23) deal with the case when the small drone was previously launched (not able to be launched at node *i* again) and has not returned to the node *j* where the hybrid truck is scheduled to serve its customer.

$$la_j^{kd,k} \geq -M(2 - x_{i,j}^k + \sum_{a\in C}\sum_{b\in C} y_{a,b,j}^{kd,k})\ ; \forall i, \forall j \in C,\ \forall k \in K, \forall kd \in KD \tag{24}$$

$$la_j^{kd,k} \leq M(2 - x_{i,j}^k + \sum_{a\in C}\sum_{b\in C} y_{a,b,j}^{kd,k})\ ; \forall i, \forall j \in C,\ \forall k \in K, \forall kd \in KD \tag{25}$$

Constraints (24) to (25) ensure that if the small drone returns to node *j* where a hybrid truck *k* serves its customer, then the auxiliary variable $la_j^{kd,k}$must be equal to 0, the state which an arc drone can leave or enter the node *j*.

$$D_j \leq Qd + M(1 - \sum_{i\in C}\sum_{p\in C} y_{i,j,p}^{kd,k}\ );\ \forall j \in C,\ \forall k \in K, \forall kd \in KD \tag{26}$$

Constraints (26) ensure that the amount of load a small drone carries to serve a customer at node *j* must be less than the drone's capacity (*QD*) in any given delivery node.

$$\sum_{i\in C} D_i \left(yt_i^k\right) + \sum_{i\in C}\sum_{kd\in KD} D_i(yd_i^{kd,k}) \leq Q; \forall k \in K \tag{27}$$

Constraints (27) enforce that the total delivery loads of both hybrid truck and small drone combined must be less than the hybrid truck capacity in any given node.

$$\sum_{i\in C} D_i\ (yvt_i^{vt}) \leq QVT; \forall vt \in VT \tag{28}$$

$$\sum_{i\in C} D_i \left(yvd_i^{vd}\right) \leq QVD; \forall vd \in VD \tag{29}$$

Constraints (28) and (29) address the similar condition as in (27) by ensuring each traditional truck *vt* must carry the load less and its capacity and so does the large drone.

The sets of constraints (30) – (31) deal with the battery consumption of the drones.

$$\tau_{i,j}^{D} + \tau_{j,p}^{D} \leq B + M(1 - \sum_{i \in C} \sum_{p \in C} y_{i,j,p}^{kd,k}); \ \forall j \in C, \forall k \in K, \forall kd \in K \tag{30}$$

Constraints (30) address that when a small drone departs from node *i*, make a delivery at node *j* and return to node *p*, it must have enough battery to cover the entire flight which must be less its capacity B.

$$bc_j^{vd} \geq bc_i^{vd} + \tau_{i,j}^{D} - M\left(1 - xvd_{i,j}^{vd}\right); \ \forall i \in C_0, \forall j \in C_+, \forall vd \in VD \tag{31}$$

$$bc_i^{vd} \leq BVD \tag{32}$$

Similarly, constraints (31) and (32) ensure that the amount of battery consumption of each large drone must be less than its battery capacity at any point in time.

$$\sum_{j \in C} y_{i,j}^{kd,k} \left(tt_i^k - dt_i^{kd,k}\right) = 0; \ \forall i \in C_0, \forall k \in K, \forall kd \in KD \tag{33}$$

$$\sum_{j \in C} y_{j,i}^{kd,k} \left(tt_i^k - dt_i^{kd,k}\right) = 0; \ \forall i \in C, \forall k \in K, \forall kd \in KD \tag{34}$$

Constraints (33) and (34) adjust the departure time and the arrival time of both small drone and hybrid truck to be the same once the two vehicles merge.

$$tt_j^k \geq tt_i^k + \tau_{i,j}^{T} - M\left(1 - x_{i,j}^{k}\right); \ \forall i \in C_0, \forall j \in C_+, \forall k \in K \tag{35}$$

$$tvt_j^{vt} \geq tvt_i^{vt} + \tau_{i,j}^{T} - M\left(1 - xvt_{i,j}^{vt}\right); \ \forall i \in C_0, \forall j \in C_+, \forall vt \in VT \tag{36}$$

$$dt_p^{kd,k} \geq dt_i^{kd,k} + \tau_{i,j}^{D} + \tau_{j,p}^{D} - M\left(1 - y_{i,j,p}^{kd,k}\right); \forall i, \forall j, \forall p \in C, \forall k \in K, \forall kd \in KD \tag{37}$$

$$tvd_j^{vd} \geq tvd_i^{vd} + \tau_{i,j}^{D} - M\left(1 - xvd_{i,j}^{vd}\right); \ \forall i \in C_0, \forall j \in C_+, \forall vd \in VD \tag{38}$$

Constraints (35), (36), (37) and (38) keep track of the arrival time of the hybrid truck, traditional truck, small drone, and large drone at every node accordingly. It adds the truck (drone) travel time to the previous customer node when the truck (drone) travels from one customer node to another customer node.

$$u_i^k - u_j^k + Q(x_{i,j}^k) \leq Q - D_j;\ \forall i,\ \forall j \in C \cup C_0 \cup C_+, \forall k \in K \tag{39}$$

$$D_i \leq u_i^k \leq Q\ ;\ \forall i,\ \forall j \in C \cup C_0 \cup C_+, \forall k \in K \tag{40}$$

$$u_i^{vt} - u_j^{vt} + Qvt(xvt_{i,j}^{vt}) \leq Qvt - D_j;\ \forall i \in N, \forall j \in N, \forall vt \in VT \tag{41}$$

$$D_i \leq u_i^{vt} \leq Qvt\ ;\ \forall i \in N, \forall j \in N, \forall vt \in VT \tag{42}$$

$$u_i^{vd} - u_j^{vd} + Qvd(xvd_{i,j}^{vd}) \leq Qvd - D_j;\ \forall i \in N, \forall j \in N, \forall vd \in VD \tag{43}$$

$$D_i \leq u_i^{vd} \leq Qvd;\ \forall i \in N, \forall j \in N, \forall vd \in VD \tag{44}$$

Pairs of constraints (39)-(40), (41)-(42), and (43)-(44) are sets of the Desrochers and Laporte (DL) sub tour elimination constraint, which ensures that there is no sub tour in all tours of the hybrid truck fleet, traditional truck fleet and large drone fleet accordingly (Desrochers & Laporte, 1991).

$$x_{i,j}^k,\ y_{i,j,p}^{kd,k}, xvt_{i,j}^{vt}, xvd_{i,j}^{vd}, yt_i^k, yd_i^{kd,k}, yvt_i^{vt}, yvd_i^{vd}, la_i^{kd,k} \in \{0,1\},$$
$$tt_j^k, dt_j^{kd,k}, bc_i^{vd},\ tvt_j^{vt}, tvd_j^{vd} \geq 0, \forall i, \forall j, \forall p \in C, \forall k \in K, \forall kd \in KD, \forall vt \in VT, \forall vd \in VD \tag{45}$$

Lastly, constraints (45) specify the types and ranges of the variables. Note that the M value in the formulation must be large enough. Thus, we can use the total time of all the delivery routes made by traditional trucks alone, i.e., solve a regular CVRP.

The I-VRPD is a generalization of the classical VRP/TSP and is thus by nature an NP-hard problem. Mixed-Integer Programming formulation was developed to obtain an optimal solution that works for the small-size problems. Because of the NP-hardness of the I-VRPD, a heuristic approach is implemented to find solutions quickly for the larger-size problems.

5.4 Solution Methods

5.4.1 Variable Neighborhood Search

5.4.1.1 Main VNS Structure

The Variable Neighborhood Search is a metaheuristic framework that was proposed by Mladenovic and Hansen (1997) for solving complex optimization problems particularly in TSP/VRP. It is based on a systematic change of the distant neighborhoods of the incumbent solution, and jumps from this solution to a new one if an improvement was made. In each neighborhood, a set of local searches will be applied repeatedly to determine the local optima. Schermer et al. implement this framework for solving the Vehicle Routing Problem with Drones heuristically. Since the concept of VRPD is equivalent to the hybrid truck operation in I-VRPD, we adapt some of the Schermer et al.'s VNS structure and modify it to fit with our problem by designing a new searching operator to explore each neighborhood.

Let denote $\mathcal{N}_k(k = 1, \dots, k_{max})$ as a finite set of pre-selected neighborhood structure and $\mathcal{N}_k(x)$ the set of solutions in the kth neighborhood of x. The following steps present the basic VNS structure:

Step 1: Select k from the set of neighborhood $\mathcal{N}_k, k = 1, \dots, k_{max}$

Step 2: Find an initial solution x

Step 3: Repeat until the stopping criteria are met. The stopping criteria can be e.g., maximum number of iterations, the CPU Time, the improvement gap between incumbent solutions, etc.

1. Set $k \leftarrow 1$
2. Repeat
3. $x' \leftarrow Shake(x, k)$; Generate a new solution x' by a random move at kth neighborhood of x ($x' \in \mathcal{N}_k(x)$);
4. $x'' \leftarrow LocalSearch(x')$; Improve the current solution by applying some local search method with x'. The new local optimum is x''
5. $x, k\,' \leftarrow NeighborhoodChange(x, x'', k)$; If the local optimum x'' is better than the incumbent, x''will become the new incumbent solution ($x' \leftarrow x''$), then $k \leftarrow 1$, else set $k \leftarrow k + 1$

The basic VNS can be extended into the Variable Neighborhood Decomposition Search (VNDS) which consists of a nested two-level VNS scheme, based on a decomposition of the optimization problem. Our problem I-VRPD is well fit with the VNDS approach by assigning two-level solutions: I-VRPD and VRP. We denote the variable y, x as a solution for I-VRPD and VRP accordingly. The basic principle is to find a solution x to a VRP as the VRP is an easier problem without the use of other vehicles, small drones, large drones, hybrid trucks, in the operation. Once the solution of VRP is found, we convert this solution into I-VRPD using the technique that will be discussed later. We iteratively explore each VRP neighborhood and generate the I-VRPD solution by solving the sub VRP problem. Different local search operators will be applied to both the VRP and I-VRPD solutions whenever the new solution is found. In addition, we further improve the I-VRPD solution by exploring a certain part of the solution. Once there is no improvement in the current I-VRPD solution, we change the VRP neighborhood and begin the new search. We repeat these steps until the stopping criteria are met. We define t_{max}, k_{max} as the maximum run time and the maximum neighborhood depth. Algorithm 1 shows the basic structure of the VNS for I-VRPD as described.

Algorithm 1. VNS (Main)

Initialize VRP solution x
$y \leftarrow ConvertVRPtoIVRPD(x)$
Repeat
    $k \leftarrow 1$
    Repeat
        $x \leftarrow Shake(x(y), k)$;
        $x' \leftarrow VRPLocalSearch(x, k)$;
        $y' \leftarrow ConvertVRPtoIVRPD(x', k)$;
        $y'' \leftarrow IVRPDImprovement(y', k)$;
        $y, x(y), k \leftarrow NeighborhoodChange(y'', y', x', k)$;
    Until $k = k_{max}$
Until $t > t_{max}$
Return y

5.4.1.2 Initialization

To quickly generate the VRP solution, we use the Clarke and Wright Saving Algorithm (Clarke & Wright, 1964), which is by far the best-known approach and yet conceptually simple, yielding reasonably good solutions to the CVRP problem (Laporte, 2009). The Saving algorithm works as follows: 1.) create n truck routes (0, i ,0) for i = 1,…, n; 2.) Compute the savings $s_{i,j}= c_{i,0} + c_{0,j} - c_{i,j}$ for i, j = 1,…, n.; 3.) Starting from the top of the savings list, merge two routes associated with the largest savings given that two delivery nodes are not in the same route, the two nodes must directly connect to the depot, e.g. $(0, j)$ and $(i, 0)$, and the total demand of the merged route must not exceed the truck capacity; 5.) repeat step (4) until no savings can be used.

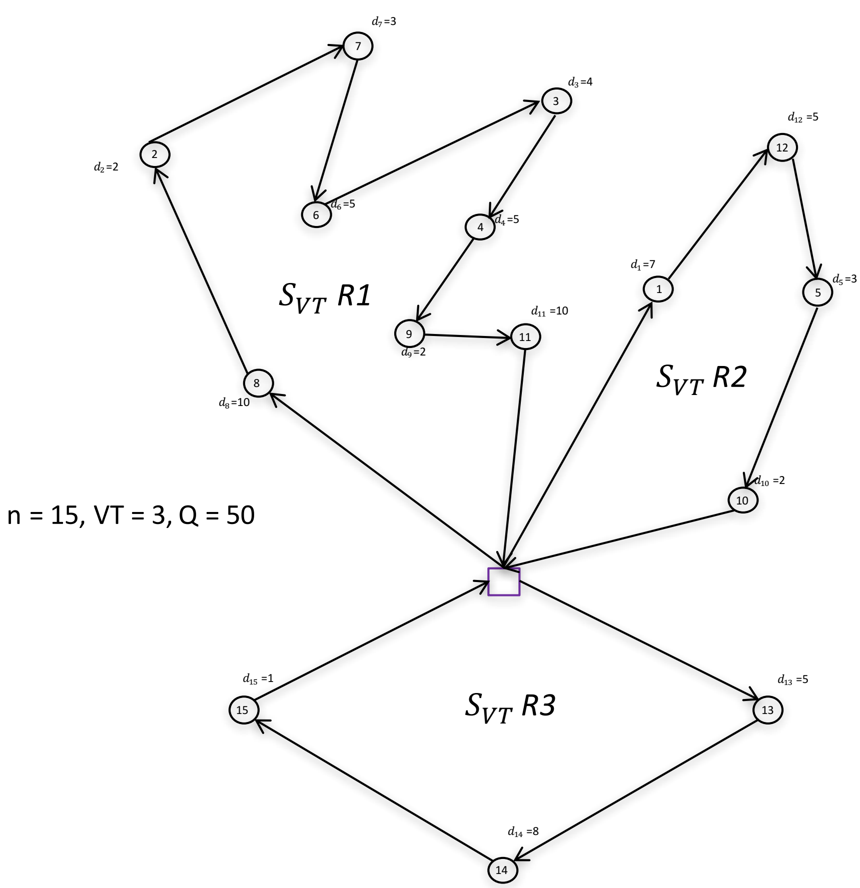


Figure 5.3. Example of VRP solution generated from Clarke and Wright Saving Algorithm

5.4.1.3 I-VRPD Route Construction from VRP Solution

This section explains how we convert the VRP solution to the I-VRPD solution as shown in line (2) and line (8) of the Algorithm1. Algorithm 2 shows the basic structure of this solution transformation. Given the total number of vehicles as V. We assign the parameters P_{VH}, P_{VD} and P_{VT} as the probabilities of the VRP routes that will be assigned to hybrid truck routes, large drone routes, and traditional truck routes in order. In other words, there are $|P_{VH} \cdot V|$ hybrid truck routes, $|P_{VD} \cdot V|$ large drone routes, and $|P_{VT} \cdot V|$ traditional truck routes. We then randomly assign each VRP route into one of the vehicle fleets. For traditional truck routes, we maintain the same solution routes as the original VRP. For each large drone route, we keep the order of customers who get served by large drones in the same order as when they get served by regular trucks and re-compute the delivery time for each route using the drone travel time. In addition, we must ensure that the following criteria are met: 1.) The total demand of the customers to be served in that route must be less than the vehicle capacity 2.) The drones' total traveling distance per trip must be less than their endurance or battery capacity for both small drones and large drones. If either one of the two criteria is violated, we remove one or more customers from a route and reinsert it to another route until the criteria are satisfied. For traditional truck routes and large drone routes, we apply some well-known local search such as 2 opt, simple relocate and swap move (Gendreau, 2008), to improve the solution quality. For the hybrid routes, we require additional steps to generate the solution that includes the small drone's delivery route as follows:

Step 1: For each hybrid truck route, select the initial truck node where the drone can be launched. The initial node is usually the node that is located closest to the depot.

Step 2: Select one of the available drones to be launched and then select one of the truck nodes as a drone delivery node using the lowest distance between the current drone node and an unvisited node.

Step 3: Remove the selected drone delivery node from the truck route and add it to the current drone route. Check the drone's battery consumption and ensure that the load does not exceed the drone's capacity. If not, reselect a new node to be served by drone.

Step 4: Select the closest truck node where the drone lands at. Check drone's battery consumption to ensure that it can fly back to a truck or else, select the new truck node for merging with a drone.

Step 5: At the current truck node, select the next available customer node for the next truck delivery from the remaining nodes that have not been previously selected to construct the drone route yet or the landing node from any drone route. If the latter is selected, we obtain a complete drone route and this drone will be available for the new selection in the next iteration. Otherwise, we can select the next customer node to be served by the truck using the closest distance to the current truck node.

Step 6: Repeat steps 2 – 5 until no drone can be selected and connect the last truck node to the depot. Complete the same steps for all hybrid truck routes.

Figure 5.4 demonstrates an example of the steps to obtain the hybrid truck route solution while Figure 5.5 shows the complete solution of the I-VRPD.

a.)

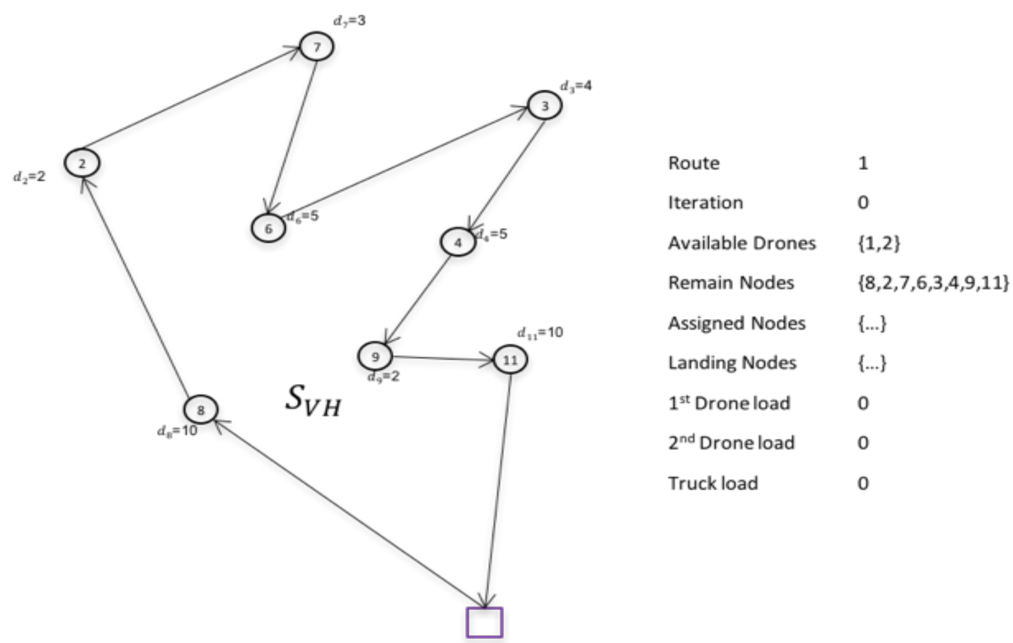
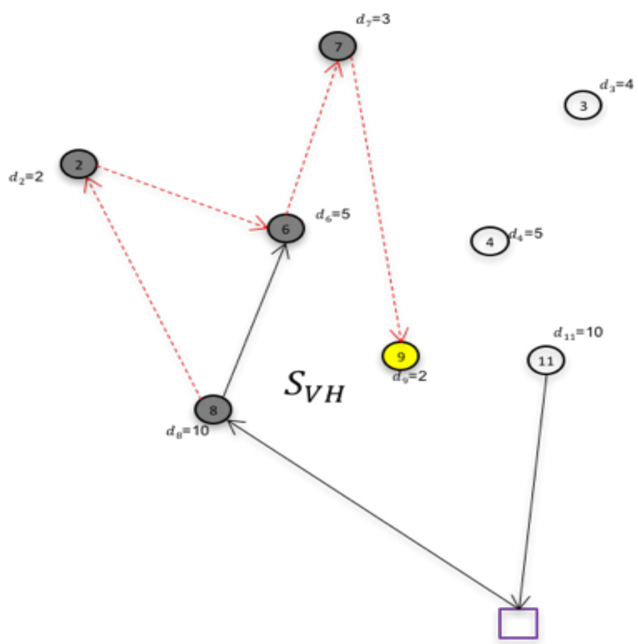


Figure 5.4 Example of S_{VH} route construction for hybrid truck. a.) Initial VRP solution for S_{VH}, b.) Construction of S_{VH} at 6th iteration, c.) Construction of S_{VH} at 10th iteration, d.) Construction of S_{VH} at 10th iteration

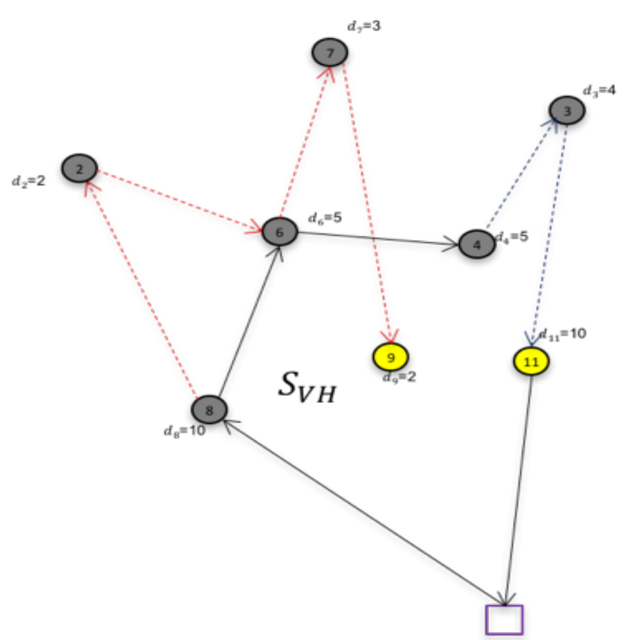
Figure 5.4 continued

b.)



Route	1
Iteration	6
Available Drones	{2}
Remain Nodes	{3,4,9,11}
Assigned Nodes	{8,2,6,7}
Landing Nodes	{9}
1st Drone load	3
2nd Drone load	0
Truck load	15

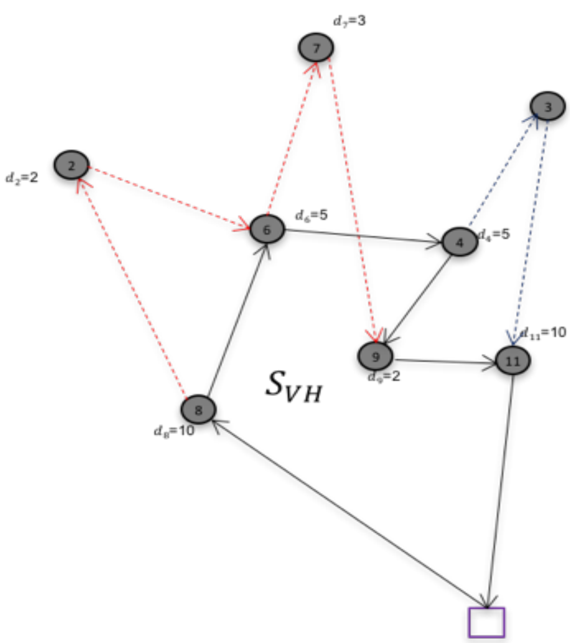
c.)



Route	1
Iteration	10
Available Drones	{...}
Remain Nodes	{9,11}
Assigned Nodes	{8,2,6,7,4,3}
Landing Nodes	{9,11}
1st Drone load	3
2nd Drone load	4
Truck load	20

Figure 5.4 continued

d.)



Route	1
Iteration	13
Available Drones	{...}
Remain Nodes	{...}
Assigned Nodes	{8,2,7,6,4,9,3,11}
Landing Nodes	{...}
1st Drone load	0
2nd Drone load	0
Truck load	32

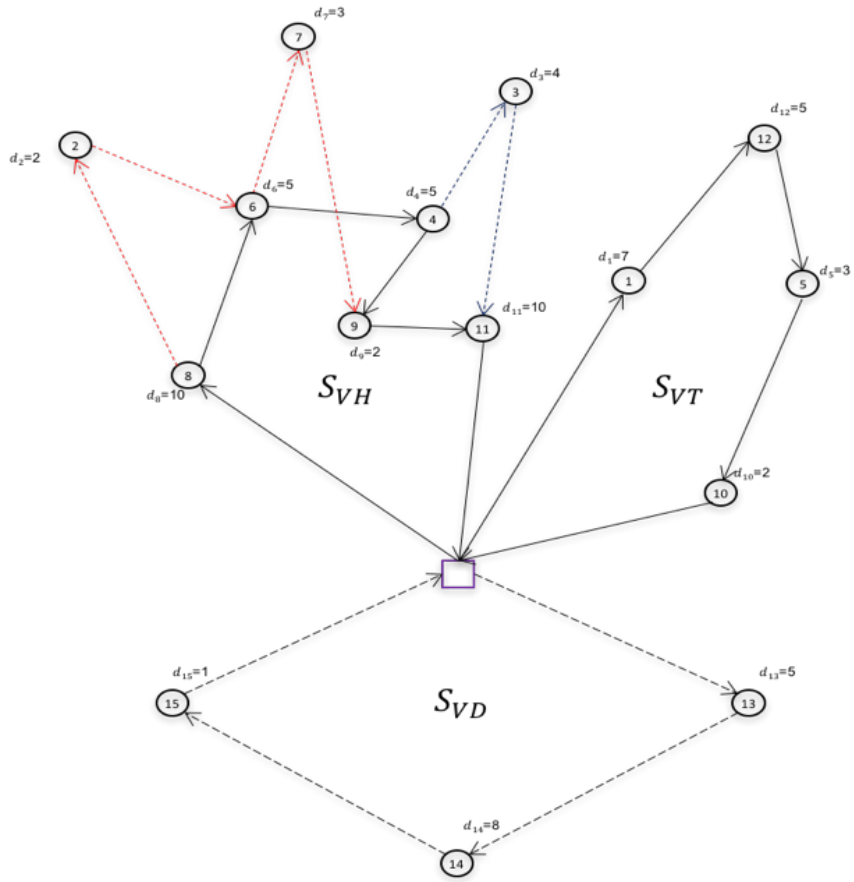


Figure 5.5 Example of complete I-VRPD route solution

Algorithm 2. ConvertVRPtoIVRPD (y, x)

Import Solution x

Assign P_{VH}, P_{VD} and P_{VT}

S_{VH} = All solution routes served by hybrid trucks*; #hybrid truck routes* = $|P_{VH} \cdot V|$

S_{VT} = All solution routes served by traditional trucks*; #traditional truck routes* = $|P_{VT} \cdot V|$

S_{VD} = All solution routes served by large drones; *#large drone routes* = $|P_{VD} \cdot V|$

Check if endurance / capacity is violated; **else** generate new S_{VD}

For all routes in S_{VH}

Set *RemainN* = *{All nodes in Route}*

Select the initial truck node

While *RemainN* $\notin \emptyset$

Select a node to be served by drone from *RemainN*

12. **If** *Drone's load* ≥ Drone Capacity || *Bat consumption* ≥ Bat capacity
13. Repeat (11)
14. **Else**
15. Update *RemainN = RemainN – {Drone Delivery Node}*
16. Select a truck node for a drone to land
17. **If** *Bat consumption* ≥ Bat capacity
18. Repeat (16)
19. **End If**
20. Update *RemainN = RemainN – {A truck node for drone to land}*
21. **End If**
22. Choose *Next Truck Delivery Node* ← min $\mathcal{D}$ from *RemainN* ∪ a truck from (16.)
23. Update *Route, RemainN = RemainN - Next Truck Delivery Node*
24. **End While**
25. **Return** S_{VH} for each *Route*
26. **End For**
27. **Return** y

5.4.1.4 Shake and VRP Local Search

This section explains the operators in line (6) and (7) of Algorithm 1. As previously mentioned, a Shake is a random move that is used to generate a new solution of a VRP solution sub-problem in the same neighborhood $x' \in \mathcal{N}_k(x)$ based on the solution of I-VRPD. This operator aims to initiate the new starting point of the search in the same neighborhood which provides a deep exploration of the solution space. The Shake operator is used after generating the initial solution of I-VRPD or a Neighborhood Change. After implementing the Shake, we apply Local Search to the new solution to improve its objective value. It is noted that while we always accept any random solution from the Shake operator, we only accept the new current solution after the Local Search when the objective of the new solution is better than the current one.

To perform the Shake and Local Search operator, we first need to covert the hybrid truck solution (S_{VH}) into the regular VRP truck solution by removing all drones nodes from the current solution and reinsert all the nodes back to the route using Saving algorithm. The large drone solution (S_{VD}) and the traditional truck solution (S_{VT}) are kept as they are. We then apply the *Route Exchange*, where two routes from different vehicle fleets are exchanged. Within each fleet type, we apply the following operators:

- VRP Crossover: Different parts of two different routes are simultaneously exchanged.
- VRP Exchange: Two nodes from different routes are simultaneously exchanged.
- VRP Swap: Two nodes from the same route are simultaneously exchanged.
- VRP Insert: A node is removed and inserted into a different location on the same route.
- VRP 2opt: Two crossover edges are selected and reordered (swap) it so that they do not cross

Figure 5.6a.) to 5.6d.) represents different operators used in both Shake and Local Search steps.

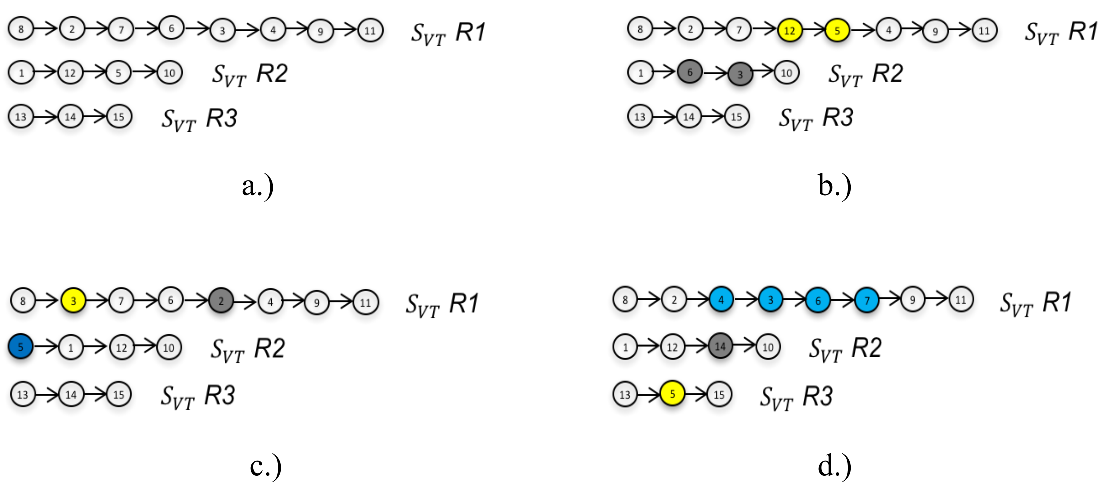


Figure 5.6 Example of different local search operators. a.) The VRP solution based on the I-VRPD solution, b.)Example of the VRP Crossover operator, c.) Example of the VRP Swap (R1) and Insert (R2), d.) Example of the VRP 2opt (R1) and Exchange (R2&R3)

5.4.1.5 I-VRPD Improvement

In this section, we perform an additional search to the hybrid truck solution (S_{VH}) by applying removing and inserting operators. There are two steps in this process: 1.) randomly remove a certain number of small drones nodes and trucks nodes from the current solution 2.) reinsert these removed nodes into the partial solution. The improved solution will only be accepted afterward. We perform this step to avoid getting stuck at the local optima as the approach to generate S_{VH}

from line (7) to (25) are fundamentally greedy and always generate the same S_{VH} of the I-VRPD solution from every VRP solution in each iteration. Algorithm 3 provides the basic structure of this search.

Algorithm 3. IVRPDImprovement(y', k);

1. **Import** S_{VH}
2. ${S_{VH}}' \leftarrow$ Remove $nx_{small\ drones}$ nodes and ny_{trucks} nodes
3. **For** All $nx_{small\ drones}$ & All $nx_{small\ drones}$
4. ${S_{VH}}'' \leftarrow$ Insert each node into ${S_{VH}}'$
5. **End For**
6. **If** ${S_{VH}}'' \leq S_{VH}$
7. $S_{VH} \leftarrow {S_{VH}}''$
8. **End If**
9. **Return** S_{VH}

5.4.2 Large Neighborhood Search

The Large Neighborhood Search (LNS) is based on a process of continual relaxation and re-optimization. An initial feasible solution of the problem is destroyed and repaired iteratively to gradually improve the solution quality. LNS offers a large move that could expand the solution search space by disintegrating a large part of the previous solution and giving the freedom to create a new one (Schrimpf et al., 2000). We modify the LNS that was previously used in multiple variants of VRP to include the certain operators that specifically work with our I-VRPD problem. Since our proposed model involves different components and constraints including different solution routes for each vehicle fleet and the sub route for drone delivery, it would be more applicable to implement the modified LNS to our problem rather than using standard local search. Algorithm 4 shows the basic structure of the proposed LNS for I-VRPD.

Algorithm 4. LNS-IVRPD

Initialize VRP solution S_{VRP}

$S_{I-VRPD} \leftarrow ConvertVRPtoIVRPD(S_{VRP})$

Generate $S_{I-VRPD}^{Best} \leftarrow S_{I-VRPD}^{Current} \leftarrow S_{I-VRPD}$

Initialize $time = 0,\ i = 0$

Repeat

  While $i \leq i_{max}$

    Enter *Destroy Phase*

      $S_{I-VRPD}^{TempD1} \leftarrow$ Call *Small drone node removal* $(S_{I-VRPD}^{Current})$

      $S_{I-VRPD}^{TempD2} \leftarrow$ Call *Hybrid truck node removal* (S_{I-VRPD}^{TempD1})

      $S_{I-VRPD}^{TempD3} \leftarrow$ Call *Large drone node removal* (S_{I-VRPD}^{TempD2})

      $S_{I-VRPD}^{TempD4} \leftarrow$ Call *Traditional truck node removal* (S_{I-VRPD}^{TempD3})

    Enter *Repair Phase*

      $S_{I-VRPD}^{TempR1} \leftarrow$ Call *Large drone node insertion* (S_{I-VRPD}^{TempD4})

      $S_{I-VRPD}^{TempR2} \leftarrow$ Call *Traditional truck node insertion* (S_{I-VRPD}^{TempD4})

      $S_{I-VRPD}^{TempR3} \leftarrow$ Call Small *drone route creation* (S_{I-VRPD}^{TempD4})

      $S_{I-VRPD}^{TempR4} \leftarrow$ Call *Hybrid truck node insertion* (S_{I-VRPD}^{TempD4})

    Select $S_{I-VRPD}^{Temp} = \min\,(S_{I-VRPD}^{TempR1}, S_{I-VRPD}^{TempR2}, S_{I-VRPD}^{TempR3}, S_{I-VRPD}^{TempR4})$

    If Cost (S_{I-VRPD}^{Temp}) < Cost $(S_{I-VRPD}^{Current})$

      $S_{I-VRPD}^{Current} \leftarrow S_{I-VRPD}^{Temp}$

      $i = 0$

    Else

      $i = i + 1$

    End If

  End While

  If Cost $(S_{I-VRPD}^{Current})$ < Cost (S_{I-VRPD}^{Best})

    $S_{I-VRPD}^{Best} \leftarrow S_{I-VRPD}^{Current}$

    $i = 0$

---

28. **Else**
29. $S_{I-VRPD}^{Current} \leftarrow S_{I-VRPD} \leftarrow ConvertVRPtoIVRPD(S_{VRP})$
30. $i = 0$
31. **End** If
32. **Until** $time \geq time_{max}$
33. **Return** S_{I-VRPD}^{Best}

The proposed algorithm based on LNS starts with similar steps that are used in VNS by generating the initial solution from VRP solution as shown in line (1) and (2). This initial solution will be stored as the current solution as well as the global best solution (line 3). At each iteration, a partial solution destruction is performed sequentially using four destroy operators on the current solution routes, which will then be repaired by the repair operators afterward (line 7 – line 16). Among all four solutions after going through the repair operators, we select the one with the lowest objective value and record it as a temporary solution (line 17). If the objective value of the temporary solution is lower than the one from the current solution, we accept the new current solution and the index i is reset to 0 (line 18-20). If this not the case, we still keep the present current solution and perform another round of destroy-repair operations to the temporary solution. We keep exploring the search space for any better solution iteratively until the index i reaches i_{max}. The current solution at the i_{max}^{th} iteration will be compared with the global best solution and be accepted as the new global best solution if it has the lower objective value (line 21-23). On the contrary, if the current solution at the i_{max}^{th} iteration has the objective value worse than the global best solution, we initialize the new S_{I-VRPD} from the VRP as the current solution. We repeat the same steps until the $time \geq time_{max}$ and return the best solution. All the different types of destroy operators and repair operators will be described in the next subsection.

5.4.2.1 Destroy Operators

Our algorithm relies on three different destroy operators, which are invoked at each iteration in sequential order. We denote p_1, p_2, p_3, and p_4 as the percentage of small drone only nodes, hybrid truck only nodes, large drone only nodes and traditional truck only nodes accordingly. The operators are presented in the sequential order of execution as follows.

Small drone node removal: This operator focuses on the hybrid truck routes by removing $\lceil p_1 \cdot C_{Only\ Samll\ Drone} \rceil$ nodes from the current hybrid truck solution (S_{VH}), with the $C_{Only\ Samll\ Drone}$ being the set of customer nodes who receive deliveries by drone only. If all nodes in a sub drone route are removed, the original launching and landing nodes are considered as truck delivery nodes and become available for launching and retrieving any available drone.

Hybrid truck node removal: This operator also focuses on the hybrid truck routes by removing $\lceil p_2 \cdot C_{Only\ Hybrid\ truck\ node} \rceil$ nodes from the current hybrid truck solution (S_{VH}), with the $C_{Only\ Hybrid\ truck\ node}$ being the set of customer nodes who receive deliveries by truck only in the hybrid truck solution (excluding the launching and landing node). If all customer nodes are removed from the hybrid truck route, the solution for that hybrid truck route becomes empty and the particular hybrid truck will not be deployed in the operation.

Large drone node removal: This operator focuses on the large drone routes by removing $\lceil p_3 \cdot C_{Large\ drone\ node} \rceil$ nodes from the current large drone solution (S_{VD}) , with the $C_{Large\ drone\ node}$ being the set of customer nodes who receive deliveries by large drone in the large drone solution. If all customer nodes are removed from the large drone route, the solution for that large drone route becomes empty and the particular large drone will not be deployed in the operation.

Traditional truck node removal: This operator focuses on the traditional truck routes by removing $\lceil p_4 \cdot C_{Traditional\ truck\ node} \rceil$ nodes from the current large drone solution (S_{VT}) , with the $C_{Traditional\ truck\ node}$ being the set of customer nodes who receive deliveries by traditional truck in the traditional truck solution. If all customer nodes are removed from the traditional truck route, the solution for that traditional truck route becomes empty and the particular truck will not be deployed in the operation.

5.4.2.2 Repair Operators

At the repair phase, each node would go through all four repair operators in parallel. Each repair operator returns the new routing solution with the selected node already inserted in the route. We compare the solutions obtained from four operators. The best-repaired solution will be selected and become the current solution of the problem. We perform the repair process until no node is left in the re-insert list and recalculate the objective value of the updated solution. The repair operators are described as follows.

Large drone node insertion: The operator takes one of the removed nodes and inserts it into any existing large drone solution route using the cheapest insertion heuristic. The new solution route must have the cumulative drone's load less than a large drone capacity and total travel distance less than its endurance.

Traditional truck node insertion: The operator takes one of the removed nodes and inserts it into any existing traditional solution route using the cheapest insertion heuristic.

Hybrid truck node insertion: The operator inserts the selected node into one of the truck routes at the current solution. The operator searches for all the feasible positions to insert the node into and selects the one with the lowest increase in total cost. If the current capacities of trucks for all routes are full, the node can be inserted into an empty route, which creates one more truck route in the solution.

Small drone route creation: The operator creates a new sub drone route by inserting the selected node between a pair of truck nodes. A pair of truck node includes a truck node where a small drone is launched from (launching node) and a node where a small drone lands at. The inserted node is a customer node served by a small drone. The operator searches for the cheapest pair among all possible combinations to construct a new drone route with the lowest increase in total cost.

Figures 5.7a.) and 5.7b.) represent the I-VRPD solution after all four destroy operators are executed and the solution after all four repair operators are executed accordingly.

a.)

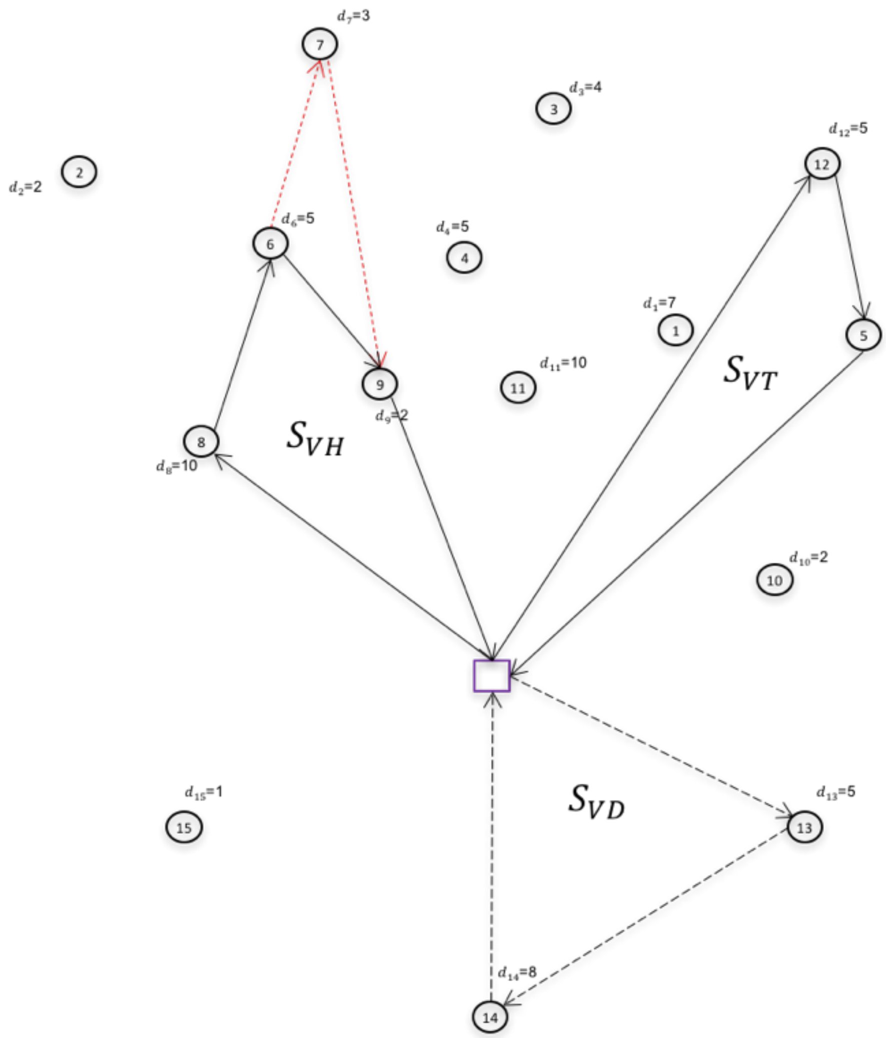
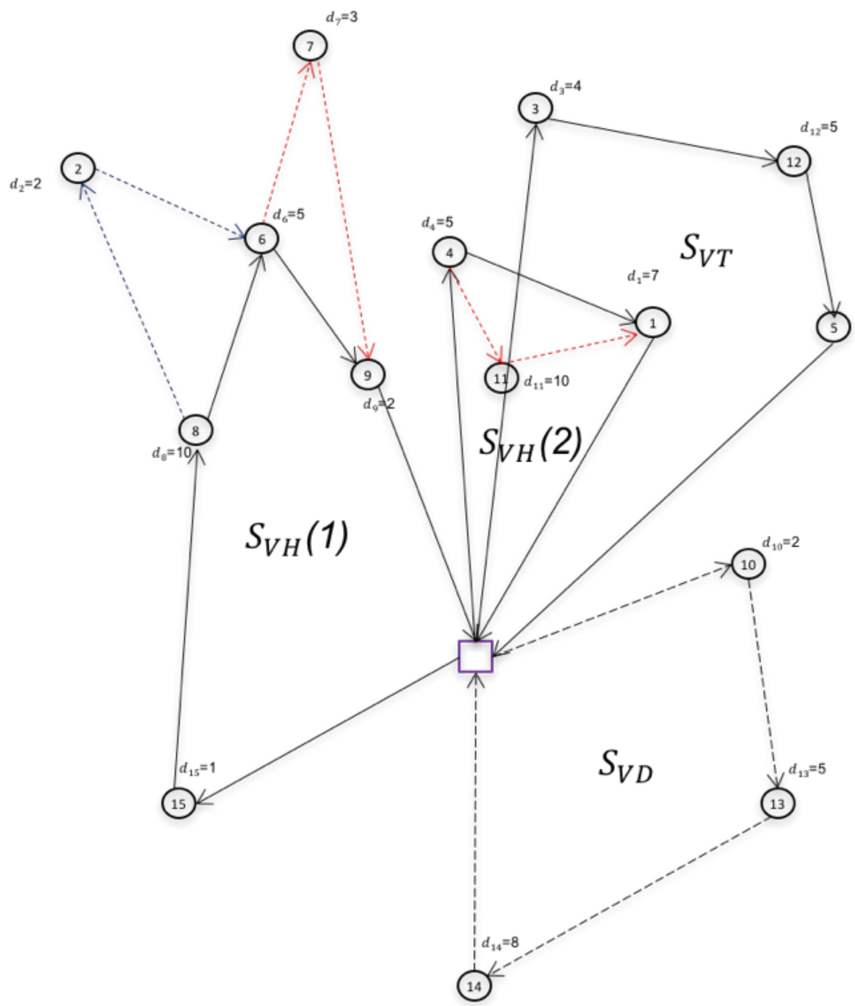


Figure 5.7 Example of LNS heuristic on I-VRPD solution. a.) Solution after the destroy operators: Small drone node route creation insertion {2,11}, Hybrid truck node insertion{1,4,15},Large drone node insertion {10},Traditional truck node insertion {3}, b.) Repair operators on 2EVRPD solution: Small Drone removal{2,3}, Hybrid truck node removal {4,11}, Large drone node removal {15}, Traditional truck node removal {1,10}.

Figure 5.7 Continued

b.)



5.5 Computational Examples and Results

This section examines the formulated MIP problem and the proposed algorithms using numerical examples and a case study. We conducted our experiments from different sets of instances, taken from four classic sets of the CVRP benchmark from Augerat et al. (sets A, B, and P) and Christofides and Elion (set E). The input data are available online at the Capacitated Vehicle Routing Problem Library (Augerat et al., 1995). We assume that the travel time can be represented by the cost metric associated with the benchmark problem. We set the truck travel time to be 1.5 time units longer than the drone travel time ($\tau^T_{j,i}$ = 1.5 $\tau^D_{i,j}$) since the drone speed is roughly about 1.5 times faster than the truck speed (Brar et al., 2015). The truck capacity and the number of trucks for each instance are excerpted from the instance input as well. Other parameters are currently assigned randomly and are subject to more calibrations in the future work. For the headers of all the tables presented in this section, we refer to *n* as the number of customer nodes, *K* as the number of hybrid trucks, *KD* as the number of small drones, *VT* as the number of traditional trucks, VD as the number of large drones, *Q* as the truck capacity, and *QD* as the small drone capacity. We assume that large drones have sufficient capacity equal to both traditional and hybrid trucks. In addition, we assume that both trucks and drones travel in Euclidean space. All the algorithms were executed in Matlab on a computer with 2.7GHz Intel Core i5 with 8GB ram RAM running Windows 7 64-bit mode. All the Mixed-Integer Linear Programming models were solved using GAMS 23.51 with CPLEX solver.

5.5.1 Experiment on the Small-Size Problems

5.5.1.1 A Case Study

In this section, we conduct a case study using the real-world scenario to investigate the usefulness of the I-VRPD model in the practical aspect and compare different solution routes under various settings. We randomly select eight customer nodes and one depot node in Lafayette/West Lafayette area. For this particular experiment, all trucks are assumed to travel in the road network while the drones travel in the air space following Euclidean distance. Other assumptions are still the same as we indicated in an earlier part of the study. We ran the MIP from Section 5.3 in the solver and generated different solution routes as shown in Figure 5.8.

a.) b.) c.) d.)

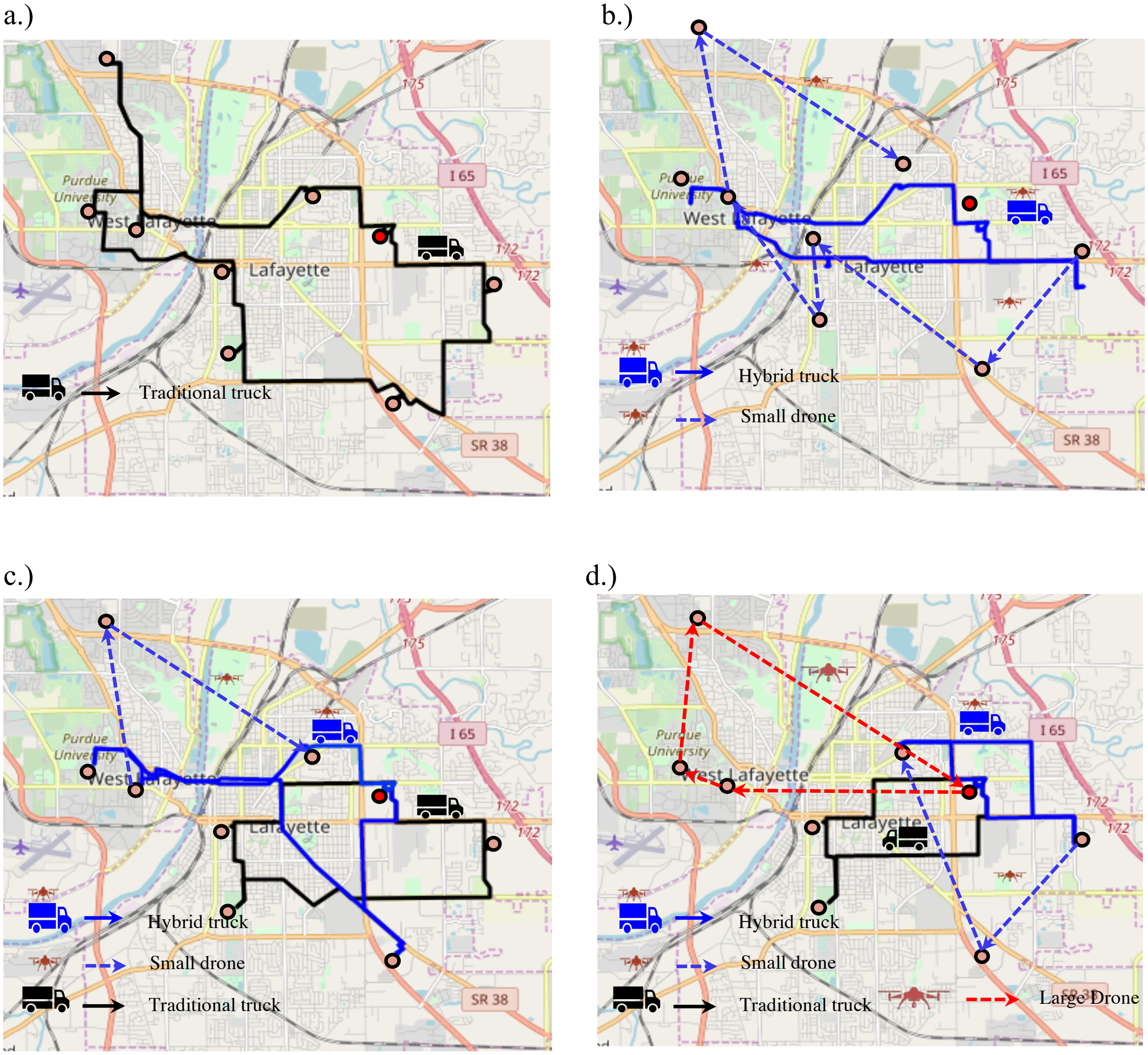


Figure 5.8 Result of the case study. (a) Single Traditional truck (Delivery time: 2046 S.). (b) Single Hybrid truck (Delivery time: 1369 S.). (c) One Hybrid truck & One Traditional truck (Delivery time: 1070 S.). (d) One Hybrid truck, One Traditional truck &One Large (Delivery time: 671 S.).

Figure 5.8a.) represents the solution route using a traditional truck alone which is the typical way of delivery and Figure 5.8b.) represents the solution route using a hybrid truck with one small drone. The gain from using a drone in the model account for 33% improvement of the delivery time. If one traditional truck is added to the operation as shown in Figure 5.8c.), it will additionally reduce the delivery time by 21.8%. Lastly, when combining a large drone, a hybrid truck and a traditional truck together in the operation, the result shows a significant reduction in delivery time by 67% from the traditional truck alone. The case study demonstrates the potential benefit of integrating different types of vehicles in last-mile delivery and illustrates the feasible solution route from the real-world scenario.

5.5.1.2 Comparison of I-VRPD MIP and other MIP Routing Models

This section compares the solution between the proposed I-VRPD and other routing models including VRPD[3] which utilizes small drones in a hybrid truck and classical CVRP on different small-size benchmark problems. We obtained the exact solutions for both models using the CPLEX solver for a comparison. The goal of this experiment is to evaluate the cost (time) saving when combining different fleet of vehicles to make deliveries. We also want to get an estimation of how long the solver would take to obtain an optimal solution for I-VRPD problem. For each instance, the number of customer nodes is set to be 8 and the maximum number of vehicles to be used is 2. The CVRP model consists of only two traditional trucks while the VRPD model consists of only a hybrid truck in which each unit is equipped with one small drone. Exactly one hybrid truck and one large drone are used in the I-VRPD. The results are shown in Table 5.1.

Table 5.1 Comparison of the results between MIP-I-VRPD, MIP-VRPD and MIP-CVRP on small-size instance

Instance	MIP CPLEX						Improvement (%)	
	I-VRPD		VRPD		CVRP			
	Objective	Runtime (Second)	Objective	Runtime (Second)	Objective	Runtime (Second)	I-VRPD v.s. VRPD	I-VRPD v.s. CVRP
A1-n8-k2	253	623.13	300	2196.09	338	88.72	15.67	25.15
A2-n8-k2	218	455.85	248	2788.66	305	75.30	12.10	28.52
A3-n8-k2	159	382.53	185	1471.09	204	95.86	14.05	22.06
B1-n8-k2	259	653.64	287	2231.58	340	76.11	9.76	23.82
B2-n8-k2	201	609.42	248	1964.14	252	64.66	18.95	20.24
P1-n8-k2	108	256.87	120	1038.87	140	40.86	10.00	22.86
P2-n8-k2	113	177.37	126	1325.49	148	36.83	10.32	23.65
Average		451.26		1859.41		68.33	**12.98**	**23.76**

[3] See the APPENDIX section for the VRPD formulation

When comparing I-VRPD to VRPD, the results show an improvement in objective value approximately by 12.98% (9.76% to 18.95%) depending on the instance. The objective value improvement is much more significant when comparing to CVRP with the average improvement of 23.77% (20.14% to 28.52%). The results from this experiment demonstrate the gain from implementing the new routing model when using all heterogeneous fleet of vehicles in the setting. In addition, it takes a significant amount of time to obtain the optimal solution for all MIP models to generate the optimal solutions even for the small-size problem (451.26 seconds for I-VRPD, 1859.41 for VRPD, and 68.33 for CVRP). Thus, we conducted more experiments to solve I-VRPD in the same instances using the proposed heuristics described in Section 5.4.

5.5.1.3 Comparison of the Proposed Heuristics and the I-VRPD MIP Model

From the previous section, we solve the I-VRPD using the MIP formulation to obtain the exact solution with the tradeoff of large computational time. In this section, we examine how well the heuristics perform when applying to the small-size instances. We ran VNS and LNS heuristics 10 times for each instance and report the Best Objective, Best GAP (%), Average Objective and Average Runtime (second) for both VNS and LNS. The GAP is computed as the percentage difference between the best (average) objective from the heuristics and the optimal solution from MIP. The goal of this experiment is to evaluate the performance of both heuristics when comparing with the exact solution from the MIP solver and to measure how fast the heuristics can generate good solutions when compared with the solver's computational time. The results are shown in Table 5.2.

Table 5.2 Comparison of the results between VNS, LNS heuristics and MIP-I-VRPD on small-size instance

Instance	I-VRPD									
	Heuristics								MIP	
	VNS	Best % Gap	Avg % GAP	Time (sec)	LNS	Best % Gap	Avg % GAP	Time (sec)	Optimal	Time (sec)
A1-n8-k2	253	0.0	0.0	52.00	253	0.0	0.0	31.23	253	623.13
A2-n8-k2	218	0.0	0.0	68.40	218	0.0	0.0	30.44	218	455.85
A3-n8-k2	159	0.0	0.0	51.76	159	0.0	0.0	30.13	159	382.53
B1-n8-k2	259	0.0	0.0	52.94	259	0.0	0.0	30.07	259	653.64
B2-n8-k2	201	0.0	0.0	53.01	201	0.0	0.0	30.13	201	609.42
P1-n8-k2	110	1.9	1.9	52.15	110	1.9	1.9	30.05	108	256.87
P2-n8-k2	113	0.0	0.0	67.20	113	0.0	0.0	30.11	113	177.37
Average		**0.3**		**56.78**		**0.3**		**30.31**		**451.259**

From Table 5.2, both VNS and LNS heuristics perform well in all small-size problems while consuming significantly less computational time than the CPLEX. The Best % Gap column shows that the VNS and LNS can equivalently find optimal solutions for all instances (0% GAP) with an exception in one instance ('P2-n8-k2' with a 1.9% GAP). It is also interesting to see that the Average % GAP returns 0 for almost all instances which indicates that the heuristics can find optimal solutions in every iteration. In terms of runtime, both heuristics use significantly less computational time than CPLEX by approximately seven times and fourteen times for VNS and LNS accordingly. Thus, we conclude that both heuristics perform well in the small-size problem and will be used to solve for an approximate solution in the larger-size problem.

5.5.1.4 Comparison of VRPD-MIP Model and CVRP-MIP Model on Small-Size Problems

This section specifically compares the solutions between the proposed VRPD and the classical CVRP on small-size problems. Exact solutions are also obtained for both models using the CPLEX solver for a comparison. The objective of this experiment is to evaluate the cost (time) saving by implementing the drone's operation with a truck under the hybrid truck operation. We also want to get an estimation of how long it would take to solve the VRPD using the MIP method to obtain an optimal solution. We conducted the experiment in the two settings: the one with a bigger truck size capacity (Q) and the one with smaller trick size capacity (Q). For the first setting, one truck is sufficient enough to be used in the operation while it is required to have at least two trucks to perform a delivery in the second setting. The results are shown in Table 5.3 and Table 5.4.

Table 5.3 Computational results of the first setting: n = 8, k = 1, Q = 200, Kd = 1, Qd = 40

Instance	MIP CPLEX				Improvement (%)
	VRPD		CVRP (No Drone)		
	Solution (Optimal)	Runtime(Second)	Solution (Optimal)	Runtime (Second)	
A1-n10-k2	235	170	279	8.08	15.77
A2-n10-k2	218	406	285	7.92	23.51
A3-n10-k2	132	332	152	8.46	13.16
B1-n10-k2	236	196	265	9.33	10.94
B2-n10-k2	182	230	189	5.85	3.70
P1-n10-k2	82	460	109	6.96	24.77
P2-n10-k2	84	180	109	7.29	22.94
Average		282	Average		16.40

Table 5.4 Computational results of the second setting: n = 8, k = 2, Q = 70, Kd = 1, Qd = 40

Instance	MIP CPLEX				Improvement (%)
	VRPD		CVRP (No Drone)		
	Solution (Optimal)	Runtime(Second)	Solution (Optimal)	Runtime (Second)	
A1-n10-k2	355	1670	383	25.74	7.31
A2-n10-k2	263	829	329	20.89	20.06
A3-n10-k2	200	1621	221	26.12	9.50
B1-n10-k2	335	1273	356	21.9	5.90
B2-n10-k2	302	1424	317	35.88	4.73
P1-n10-k2	127	487.17	148	6.57	14.19
P2-n10-k2	126	540.29	149	8.16	15.44
Average		1121	Average		11.02

The results show that implementing the trucks with drones could reduce the objective values approximately by 16.40% in the first setting and 11.02% in the second setting. The results from this numerical experiment corresponds with the result from the case study we demonstrated earlier. In general, we expect the saving to be lower than the classical CVRP but varied by the location of the customer nodes. Looking at the solution objective of the VRPD-MIP on both cases, it appears that the model with one big truck capacity performs better (return lower objective value) than the one with two small trucks size even though they both held the same total capacity.

In addition, a sensitivity analysis was conducted to evaluate the impact of major parameters and components of the method. We are specifically interested in making a comparison between different number of vehicles as well as the different sizes of truck capacity to see how it would affect the performance of the VRPD solution. For the types and number of vehicle experiment, three different operational settings are evaluated including 1.) Two small trucks – zero drone; 2.) One big truck; and, 3.) One big truck – one drone. For the size of truck capacity, we experiment with three different capacities: 1.) Q = 70 (Two truck – one drone); 2.) Q = 100 (Two truck – one drone); and, 3.) Q = 200 (One large truck – one drone). Using the same benchmark instances as Section 4.2, we report the results of the sensitivity analysis in Figure 5.9 and Figure 5.10 as follows.

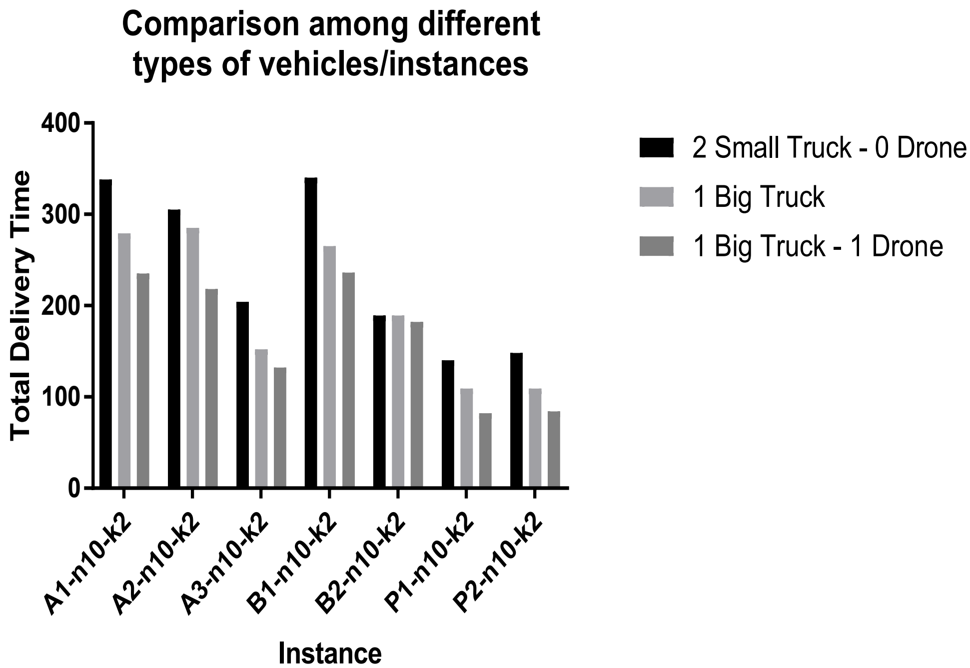


Figure 5.9 Delivery time among different types of vehicles

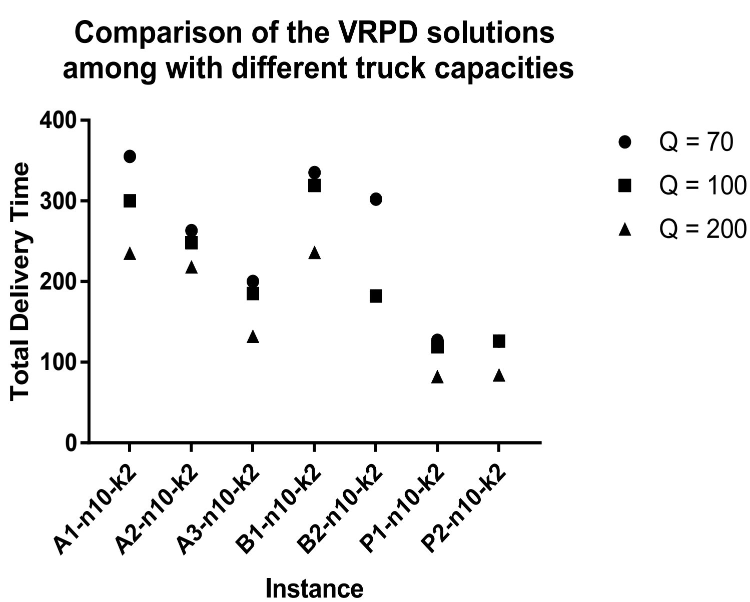


Figure 5.10 Delivery time among different truck capacities.

The result from the experiment clearly shows that the operation with one large truck and one drone outperforms other types of operations in all problem instances as shown in Figure 5.9. This demonstrates that using a drone in the operation positively results in the better objective values than using one more small truck. It is also noted that given the same total capacity, the case with one large truck returns better results than the case with the multiple small trucks as shown in Figure 5.10. We investigate this case carefully and notice that when the size of the instance is small, having an additional unit of the truck increases the total delivery time since it requires additional time for the truck to leave and return to the depot both ways. However, this might not be the case when the problem size gets larger.

5.5.2 Experiment on the Large-Size Problems

5.5.2.1 Comparison of the Proposed Heuristics and the Optimal Solution of CVRP on Various Instances

In this section, we want to test the performance of the proposed heuristics on different variations of instances. For each instance, three types of vehicles are used for the operation including a large drone, a hybrid truck and a traditional truck. At least one unit of each vehicle type must be used. A total of 46 test benchmark CVRP instances were used in this experiment and each benchmark instance was run 10 times independently. We compare the performance of our two heuristics, VNS and LNS, with the CVRP optimal solutions for each instance from the literature. We report the GAP from the CVRP solution, best solution, average solution and average runtime as shown in Table 5.5 and 5.6.

Similar to the small-size problem, it is shown that the objective of the I-VRPD is lower than the CVRP objective indicating the lower delivery cost when using the proposed model. When comparing the two heuristics, the VNS performs better than the LNS on average as the negative GAP in VNS is bigger than the one in LNS (Negative GAP on the table means better objective than the CVRP). On average, the GAP is -22.34% (-23.03%) for VNS and -19.02%(-19.53%) for LNS in the case with one small drone (two small drones). However, the VNS also consumes a larger computational time when comparing to the LNS.

Table 5.5 Results of the proposed I-VRPD heuristics and optimal CVRP on the various instances (One small drone)

Instance	CVRP (Optimal)	I-VRPD Heuristic (One small Drone)							
		VNS				LNS			
		GAP	Best	Average	Time	GAP	Best	Average	Time
A-n32-k5	784	-20.03	627.00	627.50	643.42	-16.96	651.00	655.20	64.66
A-n33-k5	661	-21.33	520.00	522.50	643.37	-18.00	542.00	548.20	62.78
A-n33-k6	742	-27.49	538.00	542.50	624.55	-24.39	561.00	561.85	65.99
A-n34-k5	778	-20.95	615.00	615.70	627.23	-20.05	622.00	630.80	64.06
A-n36-k5	799	-23.90	608.00	610.70	638.61	-22.03	623.00	623.70	72.40
A-n37-k6	949	-26.34	699.00	702.30	660.35	-24.13	720.00	728.00	67.79
A-n38-k5	730	-22.88	563.00	577.30	656.88	-15.62	616.00	621.80	65.54
A-n39-k5	822	-22.51	637.00	640.90	658.69	-19.83	659.00	682.50	65.15
A-n39-k6	831	-28.88	591.00	596.40	704.18	-23.23	638.00	643.80	74.30
A-n44-k6	937	-25.72	696.00	698.90	699.91	-20.81	742.00	755.30	70.30
A-n45-k6	944	-21.82	738.00	744.90	654.82	-18.96	765.00	776.20	72.93
A-n46-k7	914	-26.48	672.00	682.40	699.17	-22.54	708.00	714.00	71.15
A-n48-k7	1073	-23.21	824.00	830.70	722.24	-19.66	862.00	873.70	76.41
A-n53-k7	1010	-25.15	756.00	768.50	800.36	-20.99	798.00	806.50	79.87
A-n54-k7	1167	-23.22	896.00	907.10	749.17	-20.14	932.00	939.60	76.47
A-n55-k9	1073	-22.37	833.00	841.50	732.12	-18.64	873.00	896.80	74.97
A-n62-k8	1288	-23.84	981.00	1007.20	980.68	-16.38	1077.00	1084.50	80.47
A-n63-k10	1314	-20.24	1048.00	1069.50	776.59	-16.06	1103.00	1116.80	82.90
A-n64-k9	1401	-20.77	1110.00	1115.70	716.83	-18.99	1135.00	1143.40	78.12
A-n65-k9	1174	-21.64	920.00	930.20	758.00	-16.10	985.00	995.70	75.97
A-n69-k9	1159	-22.35	900.00	931.70	818.93	-20.88	917.00	931.40	81.19
B-n31-k5	672	-18.60	547.00	550.20	630.60	-18.30	549.00	555.00	64.35
B-n34-k5	788	-17.64	649.00	653.00	630.39	-16.88	655.00	659.90	62.39
B-n35-k5	955	-18.64	777.00	779.30	628.09	-17.59	787.00	795.30	63.80
B-n38-k6	805	-26.21	594.00	602.20	680.60	-20.00	644.00	658.30	71.47
B-n39-k5	549	-23.13	422.00	423.40	649.83	-21.31	432.00	437.40	66.69
B-n41-k6	829	-15.80	698.00	704.60	717.74	-13.51	717.00	721.40	71.15
E-n51-k5	521	-24.38	394.00	397.80	799.92	-22.26	405.00	426.00	85.75
E-n76-k7	682	-22.14	531.00	540.70	1009.11	-15.54	576.00	594.90	124.55
E-n76-k8	735	-24.08	558.00	565.30	1144.85	-19.73	590.00	605.10	91.45
E-n76-k10	830	-21.57	651.00	660.40	1428.96	-17.11	688.00	704.40	88.69
E-n76-k14	1021	-20.57	811.00	818.40	799.08	-17.92	838.00	849.00	81.17
P-n16-k8	450	-18.67	366.00	374.20	605.22	-19.56	362.00	362.80	60.54
P-n20-k2	216	-26.39	159.00	159.00	609.11	-25.93	160.00	160.20	60.52

Table 5.5 continued

P-n22-k2	216	-23.61	165.00	166.00	611.54	-19.91	173.00	182.60	61.13
P-n23-k8	529	-19.09	428.00	428.70	607.91	-19.09	428.00	430.20	61.37
P-n50-k7	554	-20.40	441.00	443.70	714.20	-17.87	455.00	463.30	70.45
P-n50-k8	631	-23.30	484.00	489.20	662.45	-21.55	495.00	501.80	69.71
P-n50-k10	696	-19.25	562.00	566.90	673.66	-16.81	579.00	590.40	76.98
P-n51-k10	741	-17.00	615.00	616.67	654.29	-14.71	632.00	641.55	70.46
P-n55-k10	694	-19.31	560.00	561.80	695.23	-15.99	583.00	586.40	68.48
P-n55-k15	989	-22.14	770.00	773.30	674.76	-20.32	788.00	798.27	67.19
P-n60-k10	744	-18.82	604.00	612.70	747.99	-14.78	634.00	640.80	72.26
P-n60-k15	968	-33.88	640.00	685.40	771.28	-18.18	792.00	800.40	72.42
P-n65-k10	792	-19.19	640.00	644.40	768.42	-16.04	665.00	678.50	79.76
P-n70-k10	827	-22.73	639.00	644.20	867.80	-19.47	666.00	682.78	101.48
Average		-22.34			733.68	-19.02			73.64

Table 5.6 Results of the proposed I-VRPD heuristics and optimal CVRP on the various instances (Two small drones)

Instance	CVRP (Optimal)	I-VRPD Heuristic (Two small Drones)							
		VNS				LNS			
		GAP	Best	Average	Time	GAP	Best	Average	Time
A-n32-k5	784	-20.15	626.00	628.00	641.24	-16.96	651.00	654.40	66.34
A-n33-k5	661	-23.45	506.00	509.10	656.27	-19.67	531.00	549.20	65.93
A-n33-k6	742	-28.44	531.00	536.90	663.71	-24.93	557.00	561.50	67.30
A-n34-k5	778	-21.72	609.00	611.50	665.32	-20.18	621.00	633.60	66.57
A-n36-k5	799	-26.03	591.00	596.00	752.43	-22.03	623.00	623.70	72.40
A-n37-k6	949	-26.66	696.00	700.30	778.69	-23.92	722.00	727.70	71.29
A-n38-k5	730	-25.89	541.00	555.40	756.84	-18.63	594.00	615.10	73.81
A-n39-k5	822	-24.33	622.00	624.20	738.57	-19.83	659.00	680.20	73.80
A-n39-k6	831	-29.00	590.00	592.90	741.44	-24.43	628.00	638.40	84.70
A-n44-k6	937	-26.89	685.00	689.40	771.47	-22.63	725.00	740.30	71.51
A-n45-k6	944	-23.62	721.00	736.50	749.49	-20.34	752.00	762.60	75.52
A-n46-k7	914	-27.13	666.00	675.90	725.77	-22.98	704.00	718.10	73.48
A-n48-k7	1073	-23.86	817.00	859.62	863.50	-20.04	858.00	872.80	81.09
A-n53-k7	1010	-28.02	727.00	755.38	1095.22	-20.89	799.00	805.20	82.44
A-n54-k7	1167	-23.91	888.00	902.20	1163.74	-20.57	927.00	937.40	88.06
A-n55-k9	1073	-22.83	828.00	838.60	767.92	-17.43	886.00	912.50	90.76
A-n62-k8	1288	-26.01	953.00	992.00	1195.60	-16.54	1075.00	1094.60	76.59
A-n63-k10	1314	-19.86	1053.00	1069.50	1028.11	-15.37	1112.00	1117.00	100.00
A-n64-k9	1401	-20.77	1110.00	1117.30	1491.15	-19.27	1131.00	1140.70	93.45
A-n65-k9	1174	-22.49	910.00	925.80	1224.14	-16.87	976.00	986.30	93.20
A-n69-k9	1159	-23.38	888.00	895.40	1687.10	-21.40	911.00	929.80	123.29
B-n31-k5	672	-18.60	547.00	547.10	640.39	-18.30	549.00	560.50	63.81
B-n34-k5	788	-17.64	649.00	650.20	663.54	-16.50	658.00	668.70	70.21
B-n35-k5	955	-19.37	770.00	773.40	676.91	-16.96	793.00	797.10	67.07
B-n38-k6	805	-27.08	587.00	602.30	778.33	-20.99	636.00	654.00	73.76
B-n39-k5	549	-23.13	422.00	422.90	761.76	-20.22	438.00	439.80	76.54
B-n41-k6	829	-15.92	697.00	703.10	729.71	-13.87	714.00	719.80	76.81
E-n51-k5	521	-26.49	383.00	385.40	945.47	-19.96	417.00	427.10	70.61
E-n76-k7	682	-25.37	509.00	523.50	3296.19	-18.48	556.00	580.90	126.75
E-n76-k8	735	-26.53	540.00	547.80	2306.90	-23.95	559.00	576.20	188.68
E-n76-k10	830	-23.98	631.00	650.20	2063.14	-17.71	683.00	692.20	110.25
E-n76-k14	1021	-21.06	806.00	813.71	841.43	-18.32	834.00	845.18	69.74
P-n16-k8	450	-16.89	374.00	376.20	608.13	-19.56	362.00	362.80	60.66

Table 5.6 continued

P-n20-k2	216	-27.78	156.00	156.00	613.52	-26.85	158.00	160.10	61.02
P-n22-k2	216	-27.31	157.00	157.00	631.82	-25.00	162.00	162.50	61.55
P-n23-k8	529	-19.09	428.00	428.80	612.22	-19.09	428.00	429.70	62.29
P-n50-k7	554	-21.84	433.00	437.60	798.11	-18.05	454.00	460.80	76.88
P-n50-k8	631	-24.09	479.00	484.90	700.97	-21.71	494.00	499.30	77.98
P-n50-k10	696	-19.40	561.00	567.24	730.24	-16.52	581.00	588.00	75.11
P-n51-k10	741	-17.00	615.00	617.50	680.02	-14.17	636.00	643.70	70.37
P-n55-k10	694	-20.46	552.00	556.80	720.57	-16.43	580.00	583.40	77.57
P-n55-k15	989	-21.94	772.00	773.90	682.59	-20.53	786.00	797.00	76.07
P-n60-k10	744	-18.82	604.00	610.30	782.35	-15.32	630.00	642.70	79.16
P-n60-k15	968	-19.73	777.00	780.80	810.54	-17.56	798.00	803.70	78.14
P-n65-k10	792	-20.20	632.00	636.80	884.85	-16.67	660.00	670.80	113.55
P-n70-k10	827	-25.15	619.00	624.50	1370.86	-20.68	656.00	679.20	106.26
Average		-23.03			945.40	-19.53			81.79

5.5.2.2 Comparison of the Proposed Heuristics and the Solution of VRPD on Various Instances

Continuing from the previous section, we also want to compare the results of the I-VRPD solved by the proposed heuristics with the VRPD on the same set of instances. As previously mentioned, only a hybrid truck vehicle is used in VRPD. To the best of our knowledge, we could not find any optimal VRPD solution on this set of instances in the literature. We solved the VRPD and obtained the solution using our modified LNS heuristic. The modified LNS consists of the operators which focus entirely on the route generated by the hybrid truck. We compare the performance of our two heuristics, VNS and LNS, with the best VRPD solutions for each instance which is solved 10 times. We report the GAP from the VRPD solution, best solution, average solution and average runtime in two scenarios where (1) one small drone and (2) two small drones are used as shown in Table 5.7 and 5.8.

This experiment yields similar results with the experiment in Section 5.5.2.1 in which the objective of the I-VRPD is lower than the VRPD objective as shown by the negative GAP. The GAP is, however, a bit smaller than the one with CVRP since the VRPD uses small drones in the operation while the CVRP only utilizes the regular trucks in the route. On average, the solution GAP is -

18.06% (-16.28%) for VNS and -14.57%(-12.45%) for LNS in the case with one small drone (two small drones). The results also show that the VNS outperforms the LNS using much higher computational time (bigger negative GAP).

Table 5.7 Results of the proposed I-VRPD heuristics and VRPD on the various instances (One small drone)

Instance	VRPD Solution (Best)	I-VRPD Heuristic (One small Drone)							
		VNS				LNS			
		GAP	Best	Average	Time	GAP	Best	Average	Time
A-n32-k5	750.00	-16.40	627.00	627.50	643.42	-13.20	651.00	655.20	64.66
A-n33-k5	593.00	-12.31	520.00	522.50	643.37	-8.60	542.00	548.20	62.78
A-n33-k6	653.00	-17.61	538.00	542.50	624.55	-14.09	561.00	561.85	65.99
A-n34-k5	671.00	-8.35	615.00	615.70	627.23	-7.30	622.00	630.80	64.06
A-n36-k5	717.00	-15.20	608.00	610.70	638.61	-13.11	623.00	623.70	72.40
A-n37-k6	861.00	-18.82	699.00	702.30	660.35	-16.38	720.00	728.00	67.79
A-n38-k5	654.00	-13.91	563.00	577.30	656.88	-5.81	616.00	621.80	65.54
A-n39-k5	742.00	-14.15	637.00	640.90	658.69	-11.19	659.00	682.50	65.15
A-n39-k6	758.00	-22.03	591.00	596.40	704.18	-15.83	638.00	643.80	74.30
A-n44-k6	906.00	-23.18	696.00	698.90	699.91	-18.10	742.00	755.30	70.30
A-n45-k6	902.00	-18.18	738.00	744.90	654.82	-15.19	765.00	776.20	72.93
A-n46-k7	828.00	-18.84	672.00	682.40	699.17	-14.49	708.00	714.00	71.15
A-n48-k7	998.00	-17.43	824.00	830.70	722.24	-13.63	862.00	873.70	76.41
A-n53-k7	964.00	-21.58	756.00	768.50	800.36	-17.22	798.00	806.50	79.87
A-n54-k7	1103.00	-18.77	896.00	907.10	749.17	-15.50	932.00	939.60	76.47
A-n55-k9	1031.00	-19.20	833.00	841.50	732.12	-15.32	873.00	896.80	74.97
A-n62-k8	1317.00	-25.51	981.00	1007.20	980.68	-18.22	1077.00	1084.50	80.47
A-n63-k10	1301.00	-19.45	1048.00	1069.50	776.59	-15.22	1103.00	1116.80	82.90
A-n64-k9	1266.00	-12.32	1110.00	1115.70	716.83	-10.35	1135.00	1143.40	78.12
A-n65-k9	1159.00	-20.62	920.00	930.20	758.00	-15.01	985.00	995.70	75.97
A-n69-k9	1100.00	-18.18	900.00	931.70	818.93	-16.64	917.00	931.40	81.19
B-n31-k5	662.00	-17.37	547.00	550.20	630.60	-17.07	549.00	555.00	64.35
B-n34-k5	759.00	-14.49	649.00	653.00	630.39	-13.70	655.00	659.90	62.39
B-n35-k5	914.00	-14.99	777.00	779.30	628.09	-13.89	787.00	795.30	63.80
B-n38-k6	735.00	-19.18	594.00	602.20	680.60	-12.38	644.00	658.30	71.47
B-n39-k5	518.00	-18.53	422.00	423.40	649.83	-16.60	432.00	437.40	66.69
B-n41-k6	843.00	-17.20	698.00	704.60	717.74	-14.95	717.00	721.40	71.15
E-n51-k5	490.00	-19.59	394.00	397.80	799.92	-17.35	405.00	426.00	85.75
E-n76-k7	641.00	-17.16	531.00	540.70	1009.11	-10.14	576.00	594.90	124.55
E-n76-k8	745.00	-25.10	558.00	565.30	1144.85	-20.81	590.00	605.10	91.45

Table 5.7 continued

E-n76-k10	836.00	-22.13	651.00	660.40	1428.96	-17.70	688.00	704.40	88.69
E-n76-k14	996.00	-18.57	811.00	818.40	799.08	-15.86	838.00	849.00	81.17
P-n16-k8	446.00	-17.94	366.00	374.20	605.22	-18.83	362.00	362.80	60.54
P-n20-k2	182.00	-12.64	159.00	159.00	609.11	-12.09	160.00	160.20	60.52
P-n22-k2	182.00	-9.34	165.00	166.00	611.54	-4.95	173.00	182.60	61.13
P-n23-k8	511.00	-16.24	428.00	428.70	607.91	-16.24	428.00	430.20	61.37
P-n50-k7	530.00	-16.79	441.00	443.70	714.20	-14.15	455.00	463.30	70.45
P-n50-k8	588.00	-17.69	484.00	489.20	662.45	-15.82	495.00	501.80	69.71
P-n50-k10	692.00	-18.79	562.00	566.90	673.66	-16.33	579.00	590.40	76.98
P-n51-k10	739.00	-16.78	615.00	616.67	654.29	-14.48	632.00	641.55	70.46
P-n55-k10	671.00	-16.54	560.00	561.80	695.23	-13.11	583.00	586.40	68.48
P-n55-k15	946.00	-18.60	770.00	773.30	674.76	-16.70	788.00	798.27	67.19
P-n60-k10	737.00	-18.05	604.00	612.70	747.99	-13.98	634.00	640.80	72.26
P-n60-k15	955.00	-32.98	640.00	685.40	771.28	-17.07	792.00	800.40	72.42
P-n65-k10	790.00	-18.99	640.00	644.40	768.42	-15.82	665.00	678.50	79.76
P-n70-k10	831.00	-23.10	639.00	644.20	867.80	-19.86	666.00	682.78	101.48
Average		-18.06			733.68	-14.57			73.64

Table 5.8 Results of the proposed I-VRPD heuristics and VRPD on the various instances (Two small drones)

Instance	CVRP (Optimal)	I-VRPD Heuristic (Two small Drones)							
		VNS				LNS			
		GAP	Best	Average	Time	GAP	Best	Average	Time
A-n32-k5	784	-12.32	626.00	628.00	641.24	-8.82	651.00	654.40	66.34
A-n33-k5	661	-10.76	506.00	509.10	656.27	-6.35	531.00	549.20	65.93
A-n33-k6	742	-15.85	531.00	536.90	663.71	-11.73	557.00	561.50	67.30
A-n34-k5	778	-6.45	609.00	611.50	665.32	-4.61	621.00	633.60	66.57
A-n36-k5	799	-13.34	591.00	596.00	752.43	-8.65	623.00	623.70	72.40
A-n37-k6	949	-18.79	696.00	700.30	778.69	-15.75	722.00	727.70	71.29
A-n38-k5	730	-14.67	541.00	555.40	756.84	-6.31	594.00	615.10	73.81
A-n39-k5	822	-12.89	622.00	624.20	738.57	-7.70	659.00	680.20	73.80
A-n39-k6	831	-17.71	590.00	592.90	741.44	-12.41	628.00	638.40	84.70
A-n44-k6	937	-18.36	685.00	689.40	771.47	-13.59	725.00	740.30	71.51
A-n45-k6	944	-15.57	721.00	736.50	749.49	-11.94	752.00	762.60	75.52
A-n46-k7	914	-17.16	666.00	675.90	725.77	-12.44	704.00	718.10	73.48
A-n48-k7	1073	-14.98	817.00	859.62	863.50	-10.72	858.00	872.80	81.09
A-n53-k7	1010	-23.39	727.00	755.38	1095.22	-15.81	799.00	805.20	82.44
A-n54-k7	1167	-17.47	888.00	902.20	1163.74	-13.85	927.00	937.40	88.06
A-n55-k9	1073	-17.78	828.00	838.60	767.92	-12.02	886.00	912.50	90.76
A-n62-k8	1288	-22.90	953.00	992.00	1195.60	-13.03	1075.00	1094.60	76.59
A-n63-k10	1314	-15.15	1053.00	1069.50	1028.11	-10.39	1112.00	1117.00	100.00
A-n64-k9	1401	-20.83	1110.00	1117.30	1491.15	-19.33	1131.00	1140.70	93.45
A-n65-k9	1174	-17.94	910.00	925.80	1224.14	-11.99	976.00	986.30	93.20
A-n69-k9	1159	-19.27	888.00	895.40	1687.10	-17.18	911.00	929.80	123.29
B-n31-k5	672	-17.12	547.00	547.10	640.39	-16.82	549.00	560.50	63.81
B-n34-k5	788	-13.93	649.00	650.20	663.54	-12.73	658.00	668.70	70.21
B-n35-k5	955	-14.06	770.00	773.40	676.91	-11.50	793.00	797.10	67.07
B-n38-k6	805	-18.81	587.00	602.30	778.33	-12.03	636.00	654.00	73.76
B-n39-k5	549	-17.58	422.00	422.90	761.76	-14.45	438.00	439.80	76.54
B-n41-k6	829	-15.41	697.00	703.10	729.71	-13.35	714.00	719.80	76.81
E-n51-k5	521	-15.08	383.00	385.40	945.47	-7.54	417.00	427.10	70.61
E-n76-k7	682	-20.47	509.00	523.50	3296.19	-13.13	556.00	580.90	126.75
E-n76-k8	735	-21.63	540.00	547.80	2306.90	-18.87	559.00	576.20	188.68
E-n76-k10	830	-21.61	631.00	650.20	2063.14	-15.16	683.00	692.20	110.25
E-n76-k14	1021	-18.99	806.00	813.71	841.43	-16.18	834.00	845.18	69.74
P-n16-k8	450	-16.14	374.00	376.20	608.13	-18.83	362.00	362.80	60.66
P-n20-k2	216	-8.24	156.00	156.00	613.52	-7.06	158.00	160.10	61.02

Table 5.8 continued

P-n22-k2	216	-7.65	157.00	157.00	631.82	-4.71	162.00	162.50	61.55
P-n23-k8	529	-15.75	428.00	428.80	612.22	-15.75	428.00	429.70	62.29
P-n50-k7	554	-10.91	433.00	437.60	798.11	-6.58	454.00	460.80	76.88
P-n50-k8	631	-15.67	479.00	484.90	700.97	-13.03	494.00	499.30	77.98
P-n50-k10	696	-15.38	561.00	567.24	730.24	-12.37	581.00	588.00	75.11
P-n51-k10	741	-17.45	615.00	617.50	680.02	-14.63	636.00	643.70	70.37
P-n55-k10	694	-16.62	552.00	556.80	720.57	-12.39	580.00	583.40	77.57
P-n55-k15	989	-16.99	772.00	773.90	682.59	-15.48	786.00	797.00	76.07
P-n60-k10	744	-14.81	604.00	610.30	782.35	-11.14	630.00	642.70	79.16
P-n60-k15	968	-18.04	777.00	780.80	810.54	-15.82	798.00	803.70	78.14
P-n65-k10	792	-15.85	632.00	636.80	884.85	-12.12	660.00	670.80	113.55
P-n70-k10	827	-20.95	619.00	624.50	1370.86	-16.22	656.00	679.20	106.26
Average		-16.28			945.40	-12.45			81.79

5.5.3 Discussion of the Computational Time and Heuristic Performance

To investigate the computational time of the heuristics, we used the results from Section 5.5.2.1 and Section 5.5.2.2 to plot the average runtime and the number of nodes for each instance. Figure 5.11 shows the variable number growth for different problem sizes and the time increase as problem size increases. The VNS plot shows that the runtime grows exponentially at the rate faster than the LNS plot as the number of customer nodes increases. Using the results from Table 5.5-5.8, we also plotted the mean and the standard deviation of objective values for all instances based on the delivery model as shown in Figure 5.12. It is clear that the I-VRPD problems solved by VNS and LNS return better objective values than the objective values of the traditional delivery model using trucks alone. When comparing the two heuristics, the VNS seems to perform better than the LNS with the lower mean objective. Although the VNS returns better solutions than the LNS in general, a large amount of computational time could be troublesome once the problem size gets bigger.

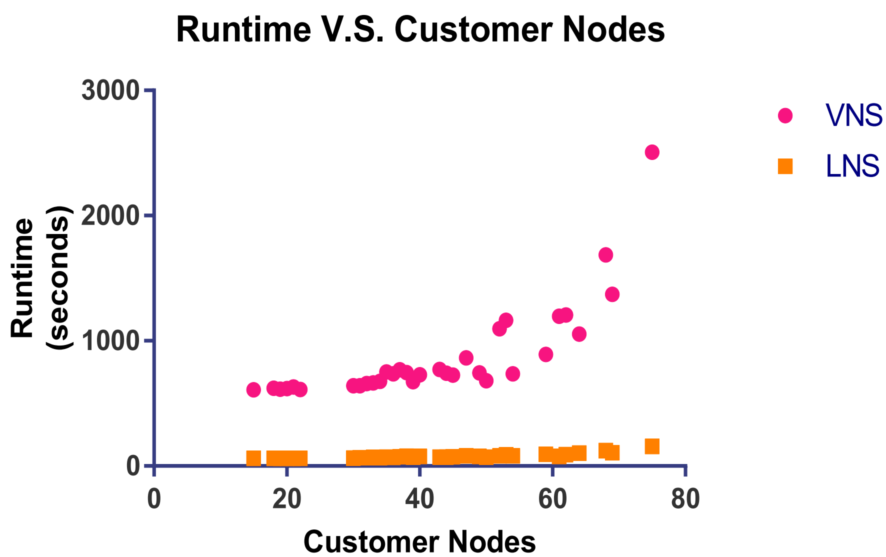


Figure 5.11 The scatter plot showing the runtime of different problem size.

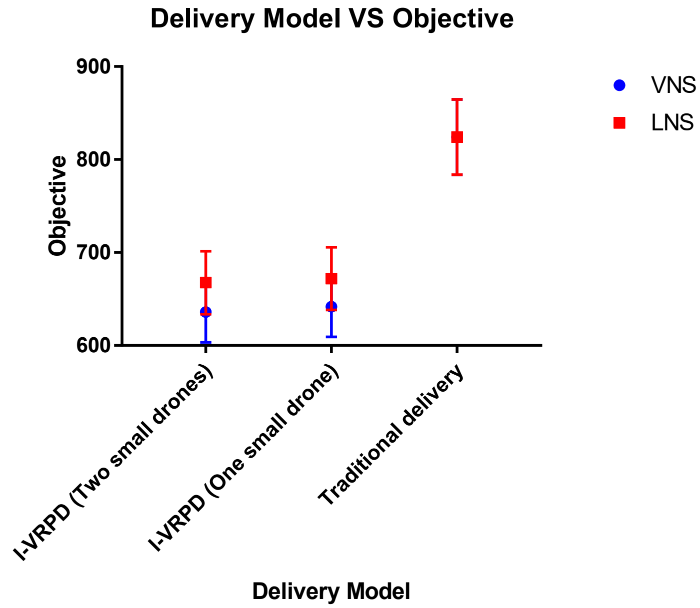


Figure 5.12 The plot showing the objective value of different delivery models.

5.6 Conclusion

In this chapter, we propose a new routing model, the Integrated Vehicle Routing Problems with Drones (I-VRPD), which combines three routing operations: Traditional truck routing, Hybrid truck routing and Large drone routing for the last-mile delivery. The I-VRPD is considered as an extension of the traditional VRP with the use of heterogeneous trucks and drones in which the hybrid truck is equipped with small drones. The MIP formulation is mathematically constructed to model the I-VRPD to solve for an optimal solution for the small-size instances, for which the results are shown in Section 5.5.1. We present different routing scenarios in the case study to visually demonstrate the implementation of the I-VRPD in the real world aspect. We additionally make a comparison of the I-VRPD and other routing models that shows the significant savings in term of delivery time. To solve the large-size instances, we propose two efficient heuristics based on the Variable Neighborhood Search (VNS) framework and Large Neighborhood Search (LNS) framework that provide optimal/near-optimal solutions while using less computational time for small-size problems. We use both heuristics to solve for the approximate solutions and compare them with the solutions from CVRP and VRPD models. Overall, through our computational study, we can show that both heuristics perform quite well and our new approach to combine different fleet vehicles clearly shows the potential benefits over the existing models in the literature.

CHAPTER 6. CONCLUSION AND FUTURE RESEARCH

6.1 Summary and Conclusion

In this dissertation, we presented an insightful study on the application of drones in last-mile delivery. We primarily focused on the truck drone routing problem with the goal to reduce the transportation time in the last-mile delivery which could potentially help companies improve their operational decisions as the e-commerce trend is growing. Contrary to conventional vehicle routing problem (VRP) that uses only one type of vehicle, a truck, to make a delivery to all customers, truck drone routing problems utilize drones in the operation to increase the delivery speed as a drone is generally faster and flexible to travel in the airspace. When combining with a truck's main advantage which is its capacity and endurance, this hybrid vehicle can outperform the traditional way of delivery. The primary application of this hybrid truck drone delivery involves improving a last-mile delivery operation with the goal of minimum delivery time of the operators which eventually would increase customer satisfaction and lead to more profit margin to the e-commerce company. The combined vehicles can also be used to deliver relief supplies to the victims in the area where traditional trucks may not be able to access. The goal is to deliver the kits to those with urgent need and primary care as quick as possible. After a careful review of the existing truck drone routing problems in the literature, we can define three main research studies which are presented in the following three chapters.

The first part of the research introduces the *multiple Traveling Salesman Problem with Drones (mTSPD).* The model is an extension of the multiple Traveling Salesman Problem (mTSP) which aims to optimally find the shortest paths for the *m* salesmen to able to visit all customer nodes exactly once. The mTSPD extends mTSP by using small drones which are attached on top of the trucks to make deliveries to some customers. We assumed that a truck behaves like a salesman and has infinite capacity to carry both drones and parcels. This problem extends the FSTSP study by deploying multiple trucks and multiple drones in the operation with the distinct feature of allowing drones to be launched and return to any available truck without restricting it to return to the one it is originally launched from. We formulated the problem as a Mixed Integer Programming (MIP) formulation to solve for the solution optimally. This problem is NP-hard and we are only able to solve small-size problems by employing the MIP model within a restricted timeframe and limited memory.

Therefore, an Adaptive Insertion algorithm (ADI) heuristic is developed to solve large-scale problems. The algorithm builds up the tours from the initial mTSP solution which are then converted to the mTSPD solution by applying removal and insertion operators. To evaluate the algorithm, we compare the solution obtained from the proposed heuristic with the optimal solution from the CPLEX solver in the small-size problems. We generated five types of problems based on the nodes distribution and depot location. The result has shown that the ADI heuristic performs very well and can reach optimal solution with significantly less time than CPLEX. In large-size instances, we tested the performance of the ADI with some of the benchmark min-max TSP/mTSP instances from TSPLIB and compare the results with the best solution from the Adapted FSTSP heuristic. We found that the algorithm returns solutions with a lower average objective when compared with the optimal objective of the truck-only operation the ones from FSTSP models both in small-size and large-size problems.

The second study considers two levels (echelons) of delivery: primary truck routing from the main depot to serve assigned customers and secondary drone routing from the truck, which behaves like a moveable intermediate depot to serve other sets of customers. We named the proposed model, the *Two Echelon Vehicle Routing Problems with Drones (2EVRPD).* The 2EVRPD can be considered as a generalized version of the mTSPD and is a variation of the classical VRP. In the proposed 2EVRPD model, we allowed multiple drones and multiple trucks to perform deliveries similar to the mTSPD model; however, there are two major distinctions to consider. First, we took into account of the capacities of both drones and trucks in the routing as well as the demand of the customers. This makes the problem much more realistic in the sense that each truck can only be equipped with certain number of drones and can carry certain number of packages.

Second, a drone itself can make multiple deliveries per launch and can carry limited number of packages. This feature creates more efficient routing solution which can be seen from different sets of experiments. The MIP formulation is mathematically constructed to optimally solve the 2EVRPD for the small-size problems. Two heuristics are developed to solve the large-size problems: Drone Truck Route Construction (DTRC) and Large Neighborhood Search (LNS). The DTRC is primarily used to generate the initial 2EVRPD solution from an empty route by first constructing the VRP solution and using the obtained VRP solution to construct the drone routes.

Once the initial 2EVRPD solution is generated, we improved the solution quality by applying the second algorithm which is based on the Large Neighborhood Search approach. This algorithm uses the well-studied destroy and repair principles by iteratively searching for a better solution using three destroy operators and three repair operators. We used these two heuristics to solve the small-size problems and compare their performances with the MIP solved by CPLEX. It appears that LNS performs significantly better than DTRC and can obtain the optimal solution in most instances. However, by limiting the amount of computational time to one hour for the medium-large instances, DTRC can return the solution using less computational time than LNS. Overall, both heuristics can outperform CPLEX solutions in all medium/large-size problems with better solution quality in LNS and better computational time in DTRC. When comparing with the CVRP from the literature, both heuristics of 2EVRPD return better objective values than the optimal CVRP objective value which demonstrates that the proposed approach provides faster delivery time than simply using trucks alone in the operation. In addition, the sensitivity analysis shows that allowing drones to make multiple deliveries prior returning to trucks significantly reduces the delivery time when comparing with the drone's route that only allows single delivery per trip.

The third research considers a novel delivery routing model called Integrated Vehicle Routing Problem with Drones (I-VRPD) which combines three vehicle fleet types: traditional truck, large drone, and hybrid truck with small drone. A large drone or cargo drone is recently introduced to the public with its main advantages to carry a large number of loads in the long distance and has a large battery capacity so that it can last a long time. A hybrid truck with small drone is defined as a truck with the equipped drone that can make a delivery one at the time and is previously used in mTSPD, and 2EVRPD. A traditional truck is merely a regular truck used for the current last-mile delivery and is used in common VRP.

The goal of this research is to develop a routing solution from these three heterogeneous fleets of vehicles and analyze its performance when comparing with the solution from a single fleet vehicle.

The problem is formulated as a Mixed Integer Programming Model (MIP) and two metaheuristics are proposed to solve this problem: The first algorithm is based on a modified Variable Neighborhood Search (VNS) framework. The algorithm begins with initializing the VRP solution and convert it to the I-VRPD solution. It will then explore the specific neighborhood through several well-known local searches and designed operator to improve the solution. When the I-VRPD solution does not improve or gets stuck in the local optima after several iterations, the algorithm alternatively explores another neighborhood and perform a new search in the new region. The algorithm repeatedly performs searching until the termination criteria is reached. The second algorithm is developed based on Large Neighborhood Search (LNS) framework similar to the one proposed in 2EVRPD model.

We used four destroy operators and four repair operators to search for the best I-VRPD solution within the reasonable time. A case study in Lafayette/West Lafayette is constructed to visually demonstrate the route solution obtained by different vehicle fleet types which appears to show that the solution route (which uses combing fleet types of vehicles) significantly performs better than a single fleet type. We also examined how well the heuristics perform by comparing the results from the heuristics with the optimal solutions obtained from the MIP. The results show that both algorithms can equivalently find optimal solutions for almost all instances on the small-size problems. For the large-size problems, we applied both VNS and LNS heuristics and compare their solutions with traditional VRP solutions (CVRP) and hybrid trucks solutions (VRPD). The results demonstrate that the proposed I-VRPD outperforms the CVRP and VRPD similar to the experiment in the small-size problem. In terms of performance between the two heuristics, VNS yields better results on average but use more computational time than LNS once the problem size gets bigger.

In the bottom line, our proposed models, mTSPD, 2EVRPD and I-VRPD extend the classical routing problem with the use of multiple drones and trucks as well as different fleet types of vehicles to make deliveries. Due to the complexity of the problem, very few studies have been conducted on the synchronization of multiple trucks and drones delivery. While the mTSPD presents more flexibility in the drone's operation as drones can be launched and retrieved by any truck, the 2EVRPD presents more realistic view as it incorporates the capacity constraints and allows multiple deliveries per trip for drones which utilize the drone's advantages more effectively. In addition to that, I-VRPD represents comprehensive truck drone routing by combining different vehicle fleet types in the last-mile delivery operation which brings out each vehicle type advantage to generate more efficient routing solutions. Throughout the entire study in this dissertation, it was shown from the numerical experiments that our proposed model, mTSPD, 2EVRPD, and I-VRPD have demonstrated the significant reduction in the delivery time compared to the classical delivery models like TSP/VRP and should be considered for the implementation in last-mile delivery.

Nevertheless, this study has several limitations. First, regarding to the comparison method between different delivery systems, we did not use the same amount of resources when deploying vehicle units in each delivery system. This could make the results much less meaningful because more vehicle units generally result in lower delivery time regardless of the types of vehicles we used. For instance, in small-size problems, only one truck is deployed to deliver all packages in the traditional delivery system while a truck with the equipped drone(s) is deployed in the proposed delivery system. Hence, it might be unfair to conclude that using hybrid truck with drone should be implemented due to lower delivery time.

One way to make this comparison fairer is to adjust the units of vehicles in each type to be the same e.g. two trucks in traditional delivery represent two vehicle units, which is equivalent to the same number of vehicle units when using one truck equipped with a drone in hybrid delivery. Since adding one more unit of truck is relatively more expensive than adding one more unit of drone, it is important to consider the cost of adding one unit of drone versus adding one unit of truck into the delivery system. This aspect can be included in the objective function, thus minimizing both cost of delivery in terms of delivery time and the monetary cost of vehicle units. Therefore, we have to design the new delivery system that measures the tradeoff between the timesaving from adding vehicles in the system and the cost of deploying vehicles in the operation.

Second, the proposed models conducted in the deterministic setting might not be practical in a real-life scenario, which all important factors such as delivery demand, weather condition, and changes in customer order can affect the system dynamically. We can resolve this limitation by having operators reevaluate these factors consistently throughout the entire operation and reinput the new information as well as execute the updated order in the vehicle to reflect the changes.

6.2 Future Research

Based on the limitations of this research, we conclude this chapter with the directions for future research:

6.2.1 Extension to the Multiple Traveling Salesman Problem with Drones (Chapter 3)

Future research could integrate other realistic constraints in the current model, such as time windows, endurance, cost, and capacity when operating drones in a different circumstance. From an algorithmic perspective, we look forward to more effective algorithms (e.g. Adaptive Large Neighborhood Search) to solve the mTSPD problem with a better objective value and lower computational time. It would be interesting to solve the model with other metaheuristic algorithms like simulated annealing or memetic algorithms.

In addition, we would like to relax the assumption that drones can only merge with trucks at the customer locations and allow both vehicles to meet at any point along the truck's path. This would make the delivery much more efficient and represent a more realistic setting. Lastly, we expect to work on developing a new Vehicle Routing Problem with Drones (VRPD) by extending the proposed mTSPD to take into account the truck capacity as well as the drone capacity.

6.2.2 Extension to the Two Echelon Vehicle Routing Problem with Drones (Chapter 4)

This research work can also be extended by using different types of both vehicles (heterogeneous fleets of both trucks and drones) as well as multiple depots to expand the range of the operation. Since the battery life of the drone is highly sensitive to its own weight and the load it carries, it would also be interesting to study how the payload (weight) and the battery consumption can affect the drone travel performance and prioritize which customer orders should get served first. Additional constraints on the weight of each item will be added and the objective function will be modified to minimize the energy consumption by a large drone which make multiple deliveries per trip.

In addition, future works may include designing the system which integrates the proposed model with drone only delivery as a unified last-mile delivery system. From an algorithmic perspective, designing other metaheuristic algorithms could be an effective way to solve the 2EVRPD problem with a better objective value and lower computational time. Considering the proposed heuristics in this research, the DTRC has the advantage of strong computation capability offsetting its worse performance when comparing to the LNS. Thus, if we develop some additional schemes inside the DTRC, this will increase the performance with little computational burden. One option is to expand the search space by introducing various search operators among different VRP routes causing the solution to get stuck in the local optimum which seems to be the case now.

Lastly, the research direction can be shifted to the logistics problem with drones which involves more strategic decisions such as the location of depots as well as both tactical and operational which include assigning customers to the opened depots and the organization of the vehicles with drones routing (LRPD).

6.2.3 Extension to the Integrated Vehicle Routing Problem with Drones (Chapter 5)

For the future work, we can consider additional fleets of vehicles or various modes of transportation in the operation including the use of electric bikes, scooters and droids. Each of these has its own advantage and would offer more alternative and efficient ways to make delivery when integrating them together rather than using the traditional one mode of transportation to perform a last-mile delivery. For example, AGVs deliver packages without human intervention and customers will be notified for the exact arrival time of the vehicle at the location. The customers are directed to pick up items from the specified locker mounted to the vehicle. Likewise, a droid is a small autonomous vehicle that can delivery packages to the doorstep by traveling only on sidewalks. They can carry loads as heavy as 50 pounds for as far as 30 miles. On the other hand, bike couriers are used for instant delivery in urban areas. They are often seen in point-to-point delivery, especially for B2B documents and prepared food. The recharging stations or transfer points can also be included to allow more flexibility for the drones to recharge and receive service before making another delivery trip. It would create more competitive advantages if we can reduce the service time or charging time for drones to overall improve the delivery performance.

Another area of improvement has to deal with the locations to deliver packages for individual customers. Due to the large size of cargo drones, it will be quite challenging for the cargo drones to land at the customer location. One possible solution is to implement a package drop mechanism which allows drone to drop off a package to the designated area at the customer location without having to land. Another alternative solution is to allocate the shared space for cargo drones to land where customers can go and pick up their packages at the certain location within the specified time frame. As for the MIP formulation, we can strengthen the computational performance of the mathematical models using valid inequalities. We can also conduct a bound analysis to investigate the maximum possible savings achievable by employing hybrid trucks and large drones in the operation. From an algorithmic perspective, designing other metaheuristic algorithms could be an effective way to solve the I-VRPD problem with a better objective value and lower computational time.

APPENDIXS A. PROBLEM DEFINITION OF VRPD

The Vehicle Routing Problems with Drones (VRPD) is a variant of a Vehicle Routing Problem with the implementation of drones in the operations. The problem aims to find the optimal set of routes for a fleet of vehicles to travel to deliver to a given set of customers. We name this problem VRPD since the problem utilizes the advantages of small (drone) and large (truck) vehicles in a delivery system. In the VRP taxonomy, the most elementary VRP considered in the literature is the one called Capacitated Vehicle Routing Problem (CVRP). The CVRP aims to determine a set of vehicles of minimum total cost over a single period with the following constraints: 1.) each route must start and end at the depot; 2.) each customer must be served by exactly one vehicle; and 3.) the total demand of each route does not exceed the vehicle capacity.

Our proposed VRPD model takes into account the capacities of both truck and drone as well as the order of launching and landing operations. In addition to covering the first main three constraints of the CVRP, the VRPD also includes the following additional constraints: 4.) each customer has to be served by exactly once either by a truck or a drone (i.e., customer demand cannot be split); 5.) the time of both truck and drone at the customer locations must be adjusted to be the same; 6.) the load that the drone can carry must be less than the drone's capacity; and 7.) drone can only be launched for the trip which it has enough battery to complete its delivery and fly back to the truck. The original VRP problem itself is already an NP-Hard and thus including the drones in the operation makes the problem much more complex and challenging as it has to decide the combinatorial solution of the truck route as well as the drone route while maintaining the order of launching and landing operations. Successful practical application of this feature in the industry could potentially bring about cost efficiency and reduce the total delivery time of the last-mile delivery.

APPENDIXS B. FORMULATION OF VRPD

The VRPD is defined on a directed graph *G = (V, E)*, where *V* is the set of *n* nodes representing customers with one depot and *E* is the set of arcs. Let $\tau^{T}_{i,j}$ be a truck travel time associated with *E,* $(i,j) \in E$ and $\tau^{D}_{i,j}$ be a drone travel time associated with *E,* $(i,j) \in E$. Differentiating the travel times for the truck and drone accounts for each vehicle's unique travel speed. The VRPD is said to be symmetric if $\tau^{T}_{i,j} = \tau^{T}_{j,i}$ and $\tau^{D}_{i,j} = \tau^{D}_{j,i}$ and asymmetric otherwise. The travel times for truck and drone matrix satisfies the triangle inequality $\tau^{T}_{i,k} + \tau^{T}_{k,j} \geq \tau^{T}_{i,j}$.

A fleet of *K* homogeneous trucks, defined as a set of *K* = {1, 2, 3, 4,…, k}, with capacity *Q* is located at the depot. The maximum number of *KD* homogeneous drones, defined as a set of *KD* = {1, 2, 3, 4,.., kd}, with the capacity *Qd* is attached along with each truck. The total length of a drone route in each launch does not exceed a pre-set limit *B* (Battery life). In our model, the fleet size of truck and the number of drones in each truck are given a priori. Denote the set of customer nodes by $C = \{1,\ 2,\ 3,\ 4,\ 5,\ 6, \ldots, n\}$. As for the depot, we assign it to two unique node numbers at $0(s)$, the starting depot, and $0(r)$, the ending depot. Set $C_0 = C \cup \{0(s)\}$ as the set of customer nodes including the starting depot and set $C_+ = C \cup \{0(r)\}$ as the set of customer nodes including the ending depot. Each customer i (i = 1, . . . , n) is associated with a known nonnegative demand, d_i , to be delivered, and the depot has a fictitious demand d_o = 0. The customer demand can be satisfied by either truck or drone delivery.

We define the following decision variables: Let $x^{k}_{i,j}$ equal to 1 if a truck *k* travels along the arc *(i, j)* ∈ *E* and 0 otherwise. This refers to the situation when the truck travels from node *i* ∈ C_0 to *j* ∈ C_+ where *i≠j.* Let $y^{kd,k}_{i,j,p}$ equal to 1 if arc *(i, j)* and *(j, p)* ∈ *E* is used on the path and 0 otherwise. This refers to the situation when a drone *kd* of truck *k* is launched from node *i* ∈ C to node *j* ∈ C (visiting customer node) and merges with a truck or the ending depot at node *p* ∈ C (recharging/swapping a battery at the truck). Let Yt^{k}_{i} equal to 1 a truck *k* serves customer node *i* and 0 otherwise. Similarly, let $Yd^{kd,k}_{i}$ equal to 1 if a drone *kd* of truck *k* serves customer node *i* and 0 otherwise.

We denote the variable Tt_j^k as the truck k arrival time at node $j \in C_+$ and $Dt_j^{kd,k}$as the drone *kd* of truck k arrival time at node $j \in C_+$. Tt_j^k and $Dt_j^{kd,k}$are the arrival times of the truck and drones at node *j* respectively. Lastly, we define other the auxiliary decision variables including 1.) u_i^k which is used in the VRP sub tour elimination constraints, 2.) $la_i^{kd,k}$which is used to indicate status whether a drone *kd* of truck *k* can be launched from node *i* or not. The proposed MIP formulation of VRPD is presented as follows.

Objective

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

The objective function (1) minimizes the total truck arrival time of trucks at the depot.

Subject to

$$\sum_{k \in K} Tt_{0(r)}^k = z \tag{2}$$

Constraint (2) straightforwardly represents the sum of each truck arrival time at the depot.

$$\sum_{k \in K} \sum_{kd \in KD} Yd_i^{kd,k} + \sum_{k \in K} Yt_i^k = 1;\ \forall i \in C \tag{3}$$

Constraints (3) ensure that each customer will receive the package either by a drone or truck. It restricts each customer to be visited exactly once by exactly one vehicle.

$$\sum_{i \in C_+} x_{0(s),i}^k = 1;\ \forall k \in K \tag{4}$$

$$\sum_{i \in C_0} x_{i,0(r)}^k = 1;\ \forall k \in K \tag{5}$$

Constraints (4) and constraints (5) impose that each truck must depart from and arrive at the depot.

$$\sum_{j \in C_+} x_{i,j}^k = \sum_{j \in C_0} x_{j,i}^k = Yt_i^k;\ \forall i \in C,\ \ \forall k \in K \tag{6}$$

Constraints (6) ensure the flow conservation of the truck route at each node *i,* which guarantees that whenever the truck *k* arrives at a node, it must depart from the node as well.

$$\sum_{\substack{i \in C \\ i \neq k}} \sum_{p \in C} y_{i,j,p}^{kd,k} \ = Yd_j^{kd,k};\ \forall j \in C, \forall k \in K, \forall kd \in KD \tag{7}$$

Constraints (7) ensure that if a drone travels from node *i* to node *j* and to node *p,* then a customer at node *j* must receive a package delivered by a drone

$$\sum_{kd \in KD} \sum_{k \in K} \sum_{\substack{j \in C \\ i \neq j}} \sum_{\substack{p \in C \\ j \neq p}} y_{i,j,p}^{kd,k} \ \leq 1;\ \forall i \in C \tag{8}$$

$$\sum_{kd \in KD} \sum_{k \in K} \sum_{\substack{p \in C \\ p \neq j}} \sum_{\substack{j \in C \\ j \neq i}} y_{p,j,i}^{kd,k} \leq 1;\ \forall i \in C \tag{9}$$

Constraints (8) and (9) ensure that only a single drone is allowed to travel from and to node *i*. Without these sets of constraints, it can occur the scenario where two drones travel from and to the node *i*.

$$2y_{i,j,p}^{kd,k} \ \leq \sum_{\substack{h \in C_0 \\ h \neq i}} x_{h,i}^k + \sum_{\substack{l \in C \\ l \neq p}} x_{l,p}^k;\ \forall i,j \ \in \ C, \forall p \ \in C, \forall k \in K, \forall kd \in KD \tag{10}$$

Constraints (10) state that a truck must visit node *i* and node *p* if the drone is launched from node *i* and is retrieved at a node.

$$\sum_{kd \in KD} \sum_{k \in K} \sum_{\substack{i \in C \\ i \neq j}} \sum_{\substack{p \in C \\ i \neq p}} y_{i,j,p}^{kd,k} \ \leq \ 1 - \sum_{kd \in KD} \sum_{k \in K} \sum_{\substack{a \in C \\ a \neq j}} \sum_{\substack{b \in C \\ a \neq b}} y_{j,a,b}^{kd,k};\ \forall j \ \in C \tag{11}$$

$$\sum_{kd\in KD}\sum_{k\in K}\sum_{\substack{i\in C\\ i\neq j}}\sum_{\substack{p\in C\\ i\neq p}} y_{i,j,p}^{kd,k} \le 1-\sum_{kd\in KD}\sum_{k\in K}\sum_{\substack{a\in C\\ a\neq j}}\sum_{\substack{b\in C\\ a\neq b}} y_{a,b,j}^{kd,k};\ \forall j\in C \tag{12}$$

Constraints (11) describe the cases that if the drone flies from node i to node j to node p, there are no other drones that make such a delivery simultaneously from node j to node a to node b. Similarly, constraints (12) describe the cases that if the drone flies from node i to node j to node p, there are no other drones that make such a delivery simultaneously from node a to node b to node j.

$$la_i^{kd,k}\left(\sum_{\substack{j\in C\\ j\neq p}}\sum_{\substack{p\in C\\ j\neq i}} y_{p,j,i}^{kd,k}\right)=0;\forall i\in N,\forall k\in K,\forall kd\in KD \tag{13}$$

$$la_i^{kd,k}\left(\sum_{\substack{j\in C\\ j\neq p}}\sum_{\substack{p\in C\\ j\neq i}} y_{i,j,p}^{kd,k}\right)=0;\forall i\in N,\ \forall k\in K,\forall kd\in KD \tag{14}$$

Constraints (13) and (14) impose that the drone is not allowed to be launched or land at the node i once the auxiliary variable $la_i^{kd,k}$ is equal to 1 and vice versa.

$$la_j^{kd,k}\ge 1-M\left(2-x_{i,j}^k-\sum_{\substack{q\in C\\ q\neq p}}\sum_{\substack{p\in C\\ q\neq i}} y_{i,q,p}^{kd,k}+la_i^{kd,k}+\sum_{\substack{a\in C\\ a\neq j}}\sum_{\substack{b\in C\\ a\neq b}} y_{a,b,j}^{kd,k}\right);\forall i,\forall j\in C,\ \forall k\in K,\forall kd\in KD \tag{15}$$

$$la_j^{kd,k}\le 1+M\left(2-x_{i,j}^k-\sum_{\substack{q\in C\\ q\neq p}}\sum_{\substack{p\in C\\ q\neq i}} y_{i,q,p}^{kd,k}+la_i^{kd,k}+\sum_{\substack{a\in C\\ a\neq j}}\sum_{\substack{b\in C\\ a\neq b}} y_{a,b,j}^{kd,k}\right);\forall i,\forall j\in C,\ \forall k\in K,\forall kd\in KD \tag{16}$$

$$la_j^{kd,k} \geq 1 - M\left(2 - x_{i,j}^k - la_i^{kd,k} + \sum_{\substack{a \in C \\ a \neq j}} \sum_{\substack{b \in C \\ a \neq b}} y_{a,b,j}^{kd,k}\right); \forall i, \forall j \in C,\ \forall k \in K, \forall kd \in KD \quad (17)$$

$$la_j^{kd,k} \leq 1 - M\left(2 - x_{i,j}^k - la_i^{kd,k} + \sum_{\substack{a \in C \\ a \neq j}} \sum_{\substack{b \in C \\ a \neq b}} y_{a,b,j}^{kd,k}\right); \forall i, \forall j \in C,\ \forall k \in K, \forall kd \in KD \quad (18)$$

Constraints (15) to (18) ensure that if the drone is launched from node *i* and has not returned at node *j,* then the auxiliary variable $la_j^{kd,k}$must be equal to 1, the state which no arc drone leaves or enters node *j*. Constraints (15) and (16) deal with the case in which the drone is launched from node *i,* and the truck travels from node *i* to node *j* at which the drone has not yet returned. Constraints (17) and (18) deal with the case when the drone was previously launched (not able to be launched at node *i* again) and has not returned to the node *j* where the truck is making a delivery at yet.

$$la_j^{kd,k} \geq -M\left(2 - x_{i,j}^k - \sum_{\substack{a \in C \\ a \neq j}} \sum_{\substack{b \in C \\ a \neq b}} y_{a,b,j}^{kd,k}\right);\ \forall i, \forall j \in C,\ \forall k \in K, \forall kd \in KD \quad (19)$$

$$la_j^{kd,k} \leq +M\left(2 - x_{i,j}^k - \sum_{\substack{a \in C \\ a \neq j}} \sum_{\substack{b \in C \\ a \neq b}} y_{a,b,j}^{kd,k}\right);\ \forall i, \forall j \in C,\ \forall k \in K, \forall kd \in KD \quad (20)$$

Constraints (19) to (20) ensure that if the drone returns to node *j* where a truck *k* serves its customer, then the auxiliary variable $la_j^{kd,k}$must be equal to 0, the state which an arc drone can leave or enter the node *j*.

$$D_j \leq Qd + M\left(1 - \sum_{\substack{i \in C \\ i \neq j}} \sum_{\substack{p \in C \\ i \neq p}} y_{i,j,p}^{kd,k}\right);\ \forall j \in C,\ \forall k \in K, \forall kd \in KD \quad (21)$$

$$\tau_{i,j}^{D} + \tau_{j,p}^{D} \leq B + M\left(1 - \sum_{\substack{i \in C \\ i \neq j}} \sum_{\substack{p \in C \\ i \neq p}} y_{i,j,p}^{kd,k}\right); \forall j \in C, \forall k \in K, \forall kd \in KD \tag{22}$$

Constraints (21) and (21) ensure drone's load must be less than its capacity and the drone's battery consumption must be less than its battery's capacity when returning to the truck node *p* accordingly.

$$\sum_{i \in C} D_i\left(Yt_i^k\right) + \sum_{i \in C} \sum_{kd \in KD} D_i(Yd_i^{kd,k}) \leq Q; \forall k \in K \tag{23}$$

Constraints (23) enforce that the total delivery loads of both truck and drone combined must be less than the truck capacity at each truck route *k*.

$$\sum_{j \in C} \sum_{p \in C} y_{i,j,p}^{kd,k} \left(Tt_i^k - Dt_i^{kd,k}\right) = 0;\ \forall i \in C_0, \forall k \in K, \forall kd \in KD \tag{24}$$

$$\sum_{j \in C} \sum_{p \in C} y_{p,j,i}^{kd,k} \left(Tt_i^k - Dt_i^{kd,k}\right) = 0;\ \forall i \in C, \forall k \in K, \forall kd \in KD \tag{25}$$

Constraints (24) state that the departure time of drones and trucks must be the same. Also, once the drone and truck are in the same node, they must wait for each other before each of them can leave the node. Similarly, constraints (25) ensure that the arrival time of both truck and drone will be the same when they merge at the same node. These sets of constraints are based on the assumption that if either the drone or truck arrives earlier than the other, the earlier one has to wait until the later one arrives (both constraints are binding, resulting in the same arrival time of both truck and drone).

$$Tt_j^k \geq Tt_i^k + \tau_{i,j}^T - M\left(1 - x_{i,j}^k\right);\ \forall i \in C_0, \forall j \in C_+, \forall k \in K \tag{26}$$

$$Dt_p^{kd,k} \geq Dt_i^{kd,k} + \tau_{i,j}^D + \tau_{j,p}^D - M\left(1 - y_{i,j,p}^{kd,k}\right); \forall i, \forall j, \forall p \in C, \forall k \in K, \forall kd \in KD \tag{27}$$

Constraints (26) keep track of the arrival time of the truck at every node. It adds the truck travel time to the previous customer node when the truck travels from one customer node to another customer node. Similarly, constraints (27) keep track of the arrival time of the drone at the node to which the drone returns after making a delivery.

$$u_i^k - u_j^k + Q(x_{i,j}^k) \leq Q - D_j;\ \forall i,\ \forall j \in C \cup C_0 \cup C_+, \forall k \in K \tag{28}$$

$$D_i \leq u_i^k \leq Q\ ;\ \forall i,\ \forall j \in C \cup C_0 \cup C_+, \forall k \in K \tag{29}$$

Constraints (28) and (29) are sets of the Desrochers and Laporte (DL) sub tour elimination constraint which ensures that there is no sub tour in all tours of the trucks.

<u>At Initial State (Time = 0)</u>

$$Tt_i^k = 0,\ Dt_j^{kd,k} = 0\ \forall i \in N, \forall j \in N, \forall k \in K, \forall kd \in KD \tag{30}$$

The set of constraints (30) set the initial departure time of drones and trucks to be zero for all nodes *i*, all drones *kd* and all trucks *k.*

$$x_{i,j}^k\ \in \{0,1\} \quad \forall i, \mathrm{j}\ \in C \cup C_0 \cup C_+ \tag{31}$$

$$y_{i,j,p}^{kd,k} \in \{0,1\} \quad \forall i, \mathrm{j}, \mathrm{p}\ \in C, \forall k \in K, \forall kd \in KD \tag{32}$$

$$Yt_i^k \in \{0,1\}\,, Yd_i^{kd,k} \in \{0,1\}\ \forall i \in C, \forall k \in K, \forall kd \in KD \tag{33}$$

$$Tt_j^k \geq 0, Dt_j^{kd,k} \geq 0,\ \forall i \in C \cup C_0 \cup C_+, \forall k \in K, \forall kd \in KD \tag{34}$$

$$la_i^{kd,k} \in \{0,1\},\ zd1_i \in \{0,1\}, zd2_i \in \{0,1\}\ \forall i \in C, \forall k \in K, \forall kd \in KD \tag{35}$$

Constraints (31–35) specify the types and ranges of the variables. Note that the M value must be large enough. We can use the total time of all the delivery routes made by trucks alone (CVRP).

REFERENCES

Adams, E. (2017). A Stronger, Faster Predator Drone Is Headed for European Airspace. https://www.wired.com/2016/12/stronger-faster-predator-drone-soon-prowl-europes-airspace/.

Agatz, N., Bouman, P., & Schmidt, M. (2018). Optimization Approaches for the Traveling Salesman Problem with Drone. *Transportation Science, 52*(4), 965-981.

Arora, K., Agarwal, S., & Tanwar, R. (2016). Solving TSP using Genetic Algorithm and Nearest Neighbour Algorithm and their Comparison. *International Journal of Scientific & Engineering Research, 7*(1), 1014-1018.

Balcik, B., Beamon, B. M., & Smilowitz, K. (2008). Last Mile Distribution in Humanitarian Relief. *Journal of Intelligent Transportation Systems, 12*(2), 51-63.

Bamburry, D. (2015). Drones: Designed for Product Delivery. *Design Management Review, 26*, 40–48.

Bektas, T. (2006). The Multiple Traveling Salesman Problem: An Overview of Formulations and Solution Procedures. *Omega, 34*(3), 209-219.

Berg, M. D. (1991). On Rectilinear Link Distance. *Computational Geometry, 1*(1), 13-34.

Bertazzi, L., Golden, B., & Wang, X. (2015). Min–Max vs. Min–Sum Vehicle Routing: A worst case analysis. *European Journal of Operational Research, 240*(2), 372-381.

Bonnington, C. (2017). President Trump Wants to Make Drone Delivery Happen. Don't Expect Much (Besides Crashes). http://www.slate.com/blogs/future_tense/2017/10/27/drone_delivery_isn't_on_its_way_even_after_president_trump_s_latest_directive.html.

Bouman, P., Agatz, N., & Schmidt, M. (2018). Dynamic Programming Approaches for the Traveling Salesman Problem with Drone. *Networks, 72*(4), 528-542.

Brar, S., Rabbat, R., Raithatha, V., Runcie, G., & Yu, A. (2015). Drones for Deliveries. *Sutardja Center for Entrepreneurship & Technology, University of California, Berkeley, Technical Report, 8*, 2015.

Breunig, U., Schmid, V., Hartl, R. F., & Vidal, T. (2016). A Large Neighborhood Based Heuristic for Two-Echelon Routing Problems. *Computers & Operations Research, 76*, 208-225.

Browne, T. (2017). Police Drones Market Increases With FAA Rules, Test Cases, Robotics Business Review. https://www.roboticsbusinessreview.com/security/police-drones-market-increases-faa-rules-test-cases/.

Buchmueller, D., Green, S., Kalyan, A., & Kimchi, G. (2016). *U.S. Patent No. US20160257401A1*. Washington, DC: U.S. Patent and Trademark Office.

Campbell, A. M., Vandenbussche, D., & Hermann, W. (2008). Routing for Relief Efforts. *Transportation Science, 42*(2), 127-145.

Campbell, J. F., Sweeney, D., & Zhang, J. (2017). Strategic Design for Delivery with Trucks and Drones. *Supply Chain Analytics Report SCMA (04 2017).*

Caric, T., & Gold, H. (2008). Vehicle Routing Problem. https://www.intechopen.com/books/vehicle_routing_problem/

Carter, A. E., & Ragsdale, C. T. (2006). A New Approach to Solving the Multiple Traveling Salesperson Problem Using Genetic Algorithms. *European Journal of Operational Research, 175*(1), 246-257.

Carlsson, J. G., & Song, S. (2018). Coordinated Logistics with a Truck and a Drone. *Management Science, 64*(9), 4052-4069.

Clarke, G. and Wright, J.R. (1964). Scheduling of Vehicle Routing Problem from a Central Depot to a Number of Delivery Points. *Operations Research, 12*, 568-581.

Cohn, P., Green, A., Langstaff M., & Roller M. (2017). Commercial Drones are Here: The Future of Unmanned Aerial Systems. https://www.mckinsey.com/industries/capital-projects-and-infrastructure/our-insights/commercial-drones-are-here-the-future-of-unmanned-aerial-systems.

Crainic, T. G., Mancini, S., Perboli, G., & Tadei, R. (2008). *Clustering-based heuristics for the two-echelon vehicle routing problem* (Vol. 46). Montreal: CIRRELT.

Crainic, T. G., Ricciardi, N., & Storchi, G. (2009). Models for Evaluating and Planning City Logistics Systems. *Transportation Science, 43*(4), 432-454.

Crainic, T. G., Mancini, S., Perboli, G., & Tadei, R. (2011, April). Multi-Start Heuristics for the Two-Echelon Vehicle Routing Problem. In *European Conference on Evolutionary Computation in Combinatorial Optimization* (pp. 179-190). Springer, Berlin, Heidelberg.

Cheng, C., Adulyasak, Y., & Rousseau, L. M. (2018). Formulations and Exact Algorithms for Drone Routing Problem. Working Paper. https://www.cirrelt.ca/DocumentsTravail/CIRRELT-2018-31.pdf.

China Is on the Fast Track to Drone Deliveries. (2018, July 03). https://www.bloomberg.com/news/features/2018-07-03/china-s-on-the-fast-track-to-making-uav-drone-deliveries.

Cuda, R., Guastaroba, G., & Speranza, M. G. (2015). A Survey on Two-Echelon Routing Problems. *Computers & Operations Research, 55*, 185-199.

Dantzig G, Fulkerson R and Johnson S. (1954) Solution of a Large Scale Traveling Salesman Problem. *Journal of the Operations Research Society of America*. *2*(4):93–410.

Dayarian, I., Savelsbergh, M., & Clarke, J. P. (2020). Same-day Delivery with drone resupply. *Transportation Science*, *54*(1), 229-249.

Desjardins, J. (2018). Is the Future of Ecommerce in Drone Deliveries? https://www.visualcapitalist.com/ecommerce-drone-deliveries/.

Desrochers, M., & Laporte, G. (1991). Improvements and Extensions to the Miller-Tucker-Zemlin Subtour Elimination Constraints. *Operations Research Letters, 10*(1), 27-36.

Dorling, K., Heinrichs, J., Messier, G. G., & Magierowski, S. (2017). Vehicle Routing Problems for Drone Delivery. *IEEE Transactions on Systems, Man, and Cybernetics: Systems, 47*(1), 70-85.

Dorr L. & Duquette, A. (2016). Press Release – DOT and FAA Finalize Rules for Small Unmanned Aircraft Systems. https://www.faa.gov/news/press releases/news_story.cfm?newsId=20515.

Drexl, M. (2012). Synchronization in Vehicle Routing—A Survey of VRPs with Multiple Synchronization Constraints. *Transportation Science, 46*(3), 297-316.

Drone Delivery: The Future of e-commerce. (2019, September). https://www.understandingecommerce.com/drone-delivery-the-future-of-ecommerce/.

E-Commerce and Emerging Logistics Technology Research Report. (2009, December). https://www.portlandoregon.gov/transportation/article/751003.

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.

Gentry, N., Hsieh, R., & Nguyen, L. (2016). *U.S. Patent No. 9527605*. Washington, DC: U.S. Patent and Trademark Office.

Glaser, A. (2017). Watch Amazon's Prime Air Make Its First Public U.S. Drone Delivery, https://www.recode.net/2017/3/24/15054884/amazon-prime-air-public-us-drone-delivery.

Global Retail E-Commerce Market Size 2014-2021. (n.d.). https://www.statista.com/statistics/379046/worldwide-retail-e-commerce-sales/

Gendreau, M., Hertz, A., & Laporte, G. (1992). New Insertion and Post Optimization Procedures for the Traveling Salesman Problem. *Operations Research, 40*(6), 1086-1094.

Gendreau, M., Iori, M., Laporte, G., & Martello, S. (2008). A Tabu Search Heuristic for the Vehicle Routing Problem with Two-Dimensional Loading Constraints. Networks: *An International Journal, 51*(1), 4-18.

Gonzalez-Feliu, J., Perboli, G., Tadei, R., & Vigo, D. (2008). The Two-Echelon Capacitated Vehicle Routing Problem. Working Paper. https://halshs.archives-ouvertes.fr/halshs-00879447/document.

Goel, A., & Kok, L. (2012). Truck Driver Scheduling in the United States. *Transportation Science,46*(3), 317-326.

Gross, D. (2013). Amazon's Drone Delivery: How Would It Work? http://www.cnn.com/2013/12/02/tech/innovation/amazon-drones-questions/index.html.

Ha, Q., Deville, Y., Pham, Q., & Hà, M. (2018). On the Min-Cost Traveling Salesman Problem with Drone. *Transportation Research Part C, 86*, 597-621.

Ham, A. M. (2018). Integrated Scheduling of M-Truck, M-Drone, and M-Depot Constrained by Time-Window, Drop-Pickup, and M-Visit Using Constraint Programming. *Transportation Research Part C: Emerging Technologies, 91*, 1-14.

Hawkins, A. J. (2019). Elroy Air's Massive Cargo Drone Completes Its First Test Flight. https://www.theverge.com/2019/8/28/20812547/elroy-air-cargo-drone-delivery-test-flight-camp-roberts-san-francisco.

Hawkins, A. J. (2019). Thousands of Autonomous Delivery Robots Are about to Descend on US College Campuses. https://www.theverge.com/2019/8/20/20812184/starship-delivery-robot-expansion-college-campus.

Heath, N. (2015). The Long-Range Drone That Can Keep Up with a Car and Fly for an Hour, https://www.techrepublic.com/blog/european-technology/the-long-range-drone-that-can-keep-up-with-a-car-and-fly-for-an-hour/.

Hern, A. (2014). DHL Launches First Commercial Drone 'Parcelcopter' Delivery Service, https://www.theguardian.com/technology/2014/sep/25/german-dhl-launches-first commercial-drone-delivery-service.

Hemmelmayr, V. C., Cordeau, J. F., & Crainic, T. G. (2012). An Adaptive Large Neighborhood Search Heuristic for Two-Echelon Vehicle Routing Problems Arising in City Logistics. *Computers & operations research, 39*(12), 3215-3228.

Hewitt, G. (2019). The Future of Logistics is Racing toward the Last Mile. https://www.mhlnews.com/transportation-distribution/article/22055503/the-future-of-logistics-is-racing-toward-the-last-mile.

Hong, I., Kuby, M., & Murray, A. (2017). A Deviation Flow Refueling Location Model for Continuous Space: A Commercial Drone Delivery System for Urban Areas. *Advances in Geocomputation Advances in Geographic Information Science,* 125-132.

Hughes, M. (2017). UPS Successfully Delivered a Package with a Drone. https://thenextweb.com/insider/2017/02/21/ups-successfully-delivered-package-drone/.

Huang, M., Smilowitz, K., & Balcik, B. (2011). Models for Relief Routing: Equity, Efficiency and Efficacy. *Procedia - Social and Behavioral Sciences, 17*, 416-437.

Jain, A. K. (2010). Data Clustering: 50 Years Beyond K-Means. *Pattern Recognition Letters, 31*(8), 651-666.

Jepsen, M., Spoorendonk, S., & Ropke, S. (2013). A Branch-and-Cut Algorithm for the Symmetric Two-Echelon Capacitated Vehicle Routing Problem. *Transportation Science, 47*(1), 23-37.

Jeong, H. Y., Song, B. D., & Lee, S. (2019). Truck-Drone Hybrid Delivery Routing: Payload-Energy Dependency and No-Fly Zones. *International Journal of Production Economics*, *214*, 220-233.

Joerss, M., Schroder, J., Neuhaus, F., Klink, C., & Mann, F. (2016). Parcel Delivery The Future of Last Mile. McKinsey & Company. http://www.www.mckinsey.com.

Karak, A., & Abdelghany, K. (2019). The Hybrid Vehicle-Drone Routing Problem for Pick- Up and Delivery Services. *Transportation Research Part C: Emerging Technologies, 102*, 427-449.

Keeney, T. (2019). Parcel Drone Delivery Should Supercharge E-commerce. https://ark-invest.com/research/parcel-drone-delivery#fn-64886-5.

Kim, S., & Moon, I. (2019). Traveling Salesman Problem with a Drone Station. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *49*(1), 42-52.

Kivelevitch, E. H., Cohen, K., & Kumar, M. (2013). A Binary Programming Solution to the Min-Max Multiple-Depots, Multiple Traveling Salesman Problem. *AIAA Infotech@Aerospace (I@A) Conference*.

Kitjacharoenchai, P., Min, B.-C., & Lee, S. (in press). Two Echelon Vehicle Routing Problem with Drones in Last Mile Delivery. *International Journal of Production Economics.* doi: 10.1016/j.ijpe.2019.107598

Kitjacharoenchai, P., Ventresca, M., Moshref-Javadi, M., Lee, S., Tanchoco, J. M., & Brunese, P. A. (2019). Multiple Traveling Salesman Problem with Drones: Mathematical model and heuristic approach. *Computers & Industrial Engineering*, *129*, 14-30.

Kumar, S. N., & Panneerselvam, R. (2012). A Survey on the Vehicle Routing Problem and Its Variants. *Intelligent Information Management, 4*(03), 66.

Laporte, G. (2009). Fifty Years of Vehicle Routing. *Transportation Science, 43*(4), 408-416.

Laporte, G., Toth, P., & Vigo, D. (2013). Vehicle Routing: Historical Perspective and Recent Contributions. https://link.springer.com/article/10.1007/s13676-013-0020-6.

Lee, H., Chen, Y., Gillai, B., & Rammohan (2016). Technological Disruption and Innovation in Last-Mile Delivery. https://www.gsb.stanford.edu/sites/gsb/files/publication-pdf/vcii-publication-technological-disruption-innovation-last-mile-delivery.pdf.

Li, J., Sun, Q., Zhou, M., & Dai, X. (2013). A New Multiple Traveling Salesman Problem and Its Genetic Algorithm-Based Solution. *2013 IEEE International Conference on Systems, Man, and Cybernetics*.

Lindner, M., Enright, A., & Bloomberg News. (2017, April 21). Shoppers Want Their Online Orders Faster. https://www.digitalcommerce360.com/2016/01/21/shoppers-want-their-online-orders-faster/.

Lu, Q., & Dessouky, M. M. (2006). A New Insertion-Based Construction Heuristic for Solving the Pickup and Delivery Problem with Time Windows. *European Journal of Operational Research,175*(2), 672-687.

Liu, W. Y., Lin, C. C., Chiu, C. R., Tsao, Y. S., & Wang, Q. (2014). Minimizing the Carbon Footprint for the Time-Dependent Heterogeneous-Fleet Vehicle Routing Problem with Alternative Paths. *Sustainability, 6*(7), 4658-4684.

Luo, Z., Liu, Z., & Shi, J. (2017). A Two-Echelon Cooperated Routing Problem for a Ground Vehicle and Its Carried Unmanned Aerial Vehicle. *Sensors, 17*(5), 1144.

Mathew, N., Smith, S. L., & Waslander, S. L. (2015). Planning Paths for Package Delivery in Heterogeneous Multirobot Teams. *IEEE Transactions on Automation Science and Engineering,12*(4), 1298-1308.

Marinelli, M., Caggiani, L., Ottomanelli, M., & Dellorco, M. (2018). En Route Truck–Drone Parcel Delivery for Optimal Vehicle Routing Strategies. *IET Intelligent Transport Systems, 12*(4), 253-261.

McFarland, M. (2017). U.S. Government Warms Up to Drone Delivery. http://money.cnn.com/2017/10/25/technology/business/drones-trump-local/index.html.

Meola, A. (2016). The FAA just Put up a Major Roadblock to Widespread Drone Usage. http://www.businessinsider.com/faa-wont-change-airspace-rules-for-drones-until-2019-2016-5.

Meola, A. (2017, July 18). Shop Online and Get Your Items Delivery by a Drone Delivery Service: The Future Amazon and Domino's Have Envisioned for Us. https://www.businessinsider.com/delivery-drones-market-service-2017-7.

Mladenović, N., & Hansen, P. (1997). Variable Neighborhood Search. *Computers & operations research, 24*(11), 1097-1100.

Moshref-Javadi, M., & Lee, S. (2017). Using Drones to Minimize Latency in Distribution Systems. In *IIE Annual Conference. Proceedings* (pp. 235-240). Institute of Industrial and Systems Engineers (IISE).

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.

Murray, C. C., & Raj, R. (2020). The Multiple Flying Sidekicks Traveling Salesman Problem: Parcel Delivery with Multiple Drones. *Transportation Research Part C: Emerging Technologies*, *110*, 368-398.

Nichols, G. (2016, March 28). This Was the First FAA-Approved Autonomous Drone Delivery in the U.S. https://www.zdnet.com/article/this-was-the-first-faa-approved-autonomous-drone-delivery-in-the-u-s/.

Otto, A., Agatz, N., Campbell, J., Golden, B., & Pesch, E. (2018). Optimization Approaches for Civil Applications of Unmanned Aerial Vehicles (UAVs) or Aerial Drones: A Survey. *Networks, 72*(4), 411-458.

Patil, S. (2016). Drone Delivery: A Game Changer for Logistics Space?. http://techstory.in/drone-delivery/.

Perez, S., & Kolodny, L. (2017) UPS Tests Show Delivery Drones Still Need Work. https://techcrunch.com/2017/02/21/ups-tests-show-delivery-drones-still-need-work/.

Poikonen, S., Wang, X., & Golden, B. (2017). The Vehicle Routing Problem with Drones: Extended Models and Connections. *Networks, 70*(1), 34-43.

Poikonen, S., & Golden, B. (2020). Multi-Visit Drone Routing Problem. *Computers & Operations Research, 113*, 104802.

Ponza, A. (2015) Optimization of Drone-Assisted Parcel Delivery. http://tesi.cab.unipd.it/51947/1/Andrea_Ponza_Analisi_e_ottimizzazione_nell%27uso_di_droni_ per_la_consegna_di_prodotti.pdf.

Pugliese, L. D. P., & Guerriero, F. (2017, September). Last-Mile Deliveries by Using Drones and Classical Vehicles. In *International Conference on Optimization and Decision Science* (pp. 557-565). Springer, Cham.

Ric (2017). What do China's Delivery Drones Look Like? - JD.Com Spotlight. http://unmannedcargo.org/chinese-delivery-drones/.

Rao, B., Gopi, A. G., & Maione, R. (2016). The Societal Impact of Commercial Drones. *Technology in Society, 45*, 83-90.

Reinelt, G. (1991). TSPLIB—A Traveling Salesman Problem Library. *ORSA Journal on Computing,3*(4), 376-384.

Romm, T. (2017). A New Trump Policy Could Let Amazon and Google Test More Drones in U.S. cities.https://www.recode.net/2017/10/25/16542940/amazon-google-drones-us-government-trump.

Ropke, S., & Pisinger, D. (2006). An Adaptive Large Neighborhood Search Heuristic for the Pickup and Delivery Problem with Time Windows. *Transportation science*, 40(4), 455-472.

Rose, C. (2013). Amazon's Jeff Bezos Looks to the Future. https://www.cbsnews.com /news/amazons-jeff-bezos-looks-to-the-future/.

Santos, F. A., Mateus, G. R., & da Cunha, A. S. (2014). A Branch-and-Cut-and-Price Algorithm for the Two-Echelon Capacitated Vehicle Routing Problem. *Transportation Science, 49*(2), 355-368.

Schermer, D., Moeini, M., & Wendt, O. (2018). *A Variable Neighborhood Search Algorithm for Solving the Vehicle Routing Problem with Drones (p. 33).* Technische Universität Kaiserslautern: Technical Report. https://www.researchgate.net/publication/326478770_A_Variable_Neighborhood_Search_Algorithm_for_Solving_the_Vehicle_Routing_Problem_with_Drones.

Schrimpf, G., Schneider, J., Stamm-Wilbrandt, H., & Dueck, G. (2000). Record Breaking Optimization Results Using the Ruin and Recreate Principle. *Journal of Computational Physics, 159*(2), 139-171.

Sedighpour, M., Yousefikhoshbakht, M., & Darani, N. (2011). An Effective Genetic Algorithm for Solving the Multiple Traveling Salesman Problem. *Journal of Optimization in Industrial Engineering, 8*, 73-79.

Shaw, P. (1998, October). Using Constraint Programming and Local Search Methods to Solve Vehicle Routing Problems. In *International conference on principles and practice of constraint programming* (pp. 417-431). Springer, Berlin, Heidelberg.

Shivakumar, F. (2019). Giant Cargo Drones will Deliver Packages Farther and Faster. https://www.theverge.com/2019/6/10/18657150/autonomous-cargo-drones-delivery boeing-aircraft-faa-regulation

Smith, D. (2013). Amazon Prime Air: 5 Major Weaknesses of The Proposed Drone Delivery Service. http://www.ibtimes.com/amazon-prime-air-5-major-weaknesses-proposed-drone-delivery-service-1491978.

Toth, P., & Vigo, D. (2002). An Overview of Vehicle Routing Problems. In The Vehicle Routing Problem (pp. 1-26). Society for Industrial and Applied Mathematics.

Tu, P., Dat, N., & Dung, P. (2018). Traveling Salesman Problem with Multiple Drones. In Proceedings of the Ninth International Symposium on Information and Communication Technology, 46-53. ACM.

Ulmer, M. W., & Thomas, B. W. (2017). Same-Day Delivery with a Heterogeneous Fleet of Drones and Vehicles (pp. 1-30). Technical Report. https://web.winforms.phil.tu-bs.de/paper/ulmer/Ulmer_drones.pdf.

Ungerleider, N. (2016) The Technology That Gets a Package From The Warehouse To Your House. https://www.fastcompany.com/3066534/the-technology-that-gets-a-package-from-the-warehouse-to-your-house.

Vincent, J. (2017). Google's Project Wing has Successfully Tested Its Air Traffic Control System for Drones, at: https://www.theverge.com/2017/6/8/15761220/google-project-wing-drone-air-traffic-control-tests.

Vincent, J. (2017, March 24). Watch Amazon's Prime Air Drone Make Its First Demo Delivery in the US. https://www.theverge.com/2017/3/24/15047424/amazon-prime-air-drone-delivery-public-us-test-mars.

Volkman, E. (2018, June 21). 8 Things That Can Be Delivered By Drone. https://tech.co/8-things-can-delivered-drone-2017-12.

Wang, X., Poikonen, S., & Golden, B. (2016). The Vehicle Routing Problem with Drones: Several Worst-Case Results. *Optimization Letters, 11*(4), 679-697.

Wang, Z., & Sheu, J. B. (2019). Vehicle Routing Problem with Drones. *Transportation research part B: methodological, 122*, 350-364.

Xavier, I. (2014). *Capacitated Vehicle Routing Problem Library* [Data File]. http://vrp.atd-lab.inf.puc-rio.br/index.php/en/.

Yalçındag, S., Matta, A., & Sahin, E. (2011). Human Resource Scheduling and Routing Problem in Home Health Care Context: A Literature Review. *ORAHS/Cardiff, United Kingdom.*

Yurek, E. E., & Ozmutlu, H. C. (2018). A Decomposition-Based Iterative Optimization Algorithm for Traveling Salesman Problem with Drone. T*ransportation Research Part C: Emerging Technologies, 91*, 249-262.

Zito, D. & Radocaj, L. (2016). First Test Delivery of the Workhorse Group HorseFly Drone Featured on NBC-TV Affiliate. https://globenewswire.com/news-release/2016/01/07/800262/0/en/First-Test-Delivery-of-the-Workhorse-Group-HorseFly-Drone-Featured-on-NBC-TV-Affiliate.html.

PUBLICATIONS

1. Kitjacharoenchai, P., Jeong, H. Y., & Lee, S. (2020). Integrated Vehicle Routing Problem with Drones for Last Mile Delivery. *Manuscript submitted for publication.*
2. Kitjacharoenchai, P., Min, B.-C., & Lee, S. (Accepted, in-press). Two Echelon Vehicle Routing Problem with Drones in Last Mile Delivery. *International Journal of Production Economics.* doi: 10.1016/j.ijpe.2019.107598.
3. Kitjacharoenchai, P., & Lee, S. (2019). Vehicle Routing Problem with Drones for Last Mile Delivery. *Procedia Manufacturing*, *39*(C), 314-324.
4. Kitjacharoenchai, P., Ventresca, M., Moshref-Javadi, M., Lee, S., Tanchoco, J. M., & Brunese, P. A. (2019). Multiple Traveling Salesman Problem with Drones: Mathematical Model and Heuristic Approach. *Computers & Industrial Engineering*, *129*, 14-30.