A Vehicle Routing Problem with Movement Synchronization of Drones, Sidewalk Robots, or Foot-Walkers

by

Puyuan Deng

A thesis submitted in conformity with the requirements for the degree of Master of Applied Science
Department of Civil & Mineral Engineering
University of Toronto

© Copyright by Puyuan Deng 2018
A Vehicle Routing Problem with Movement Synchronization of Drones, Sidewalk Robots, or Foot-Walkers

Puyuan Deng
Master of Applied Science
Department of Civil & Mineral Engineering
University of Toronto
2018

Abstract

The vehicle routing problem (VRP) and its variants have many city logistics applications, such as goods delivery, mail service, and people movement. The VRP extension with movement synchronization (VRPMS) has potential applications of drone and robot technologies to assist with the delivery of parcels. VRPMS seeks the optimal route for a set of composite resources, e.g. delivery van with drones, or delivery van with sidewalk robots. This paper proposes an exact formulation of the problem, and a metaheuristic approach to solve larger instances of the VRPMS in order to assess the economic benefits of the different technologies. A numerical analysis demonstrates that the sidewalk robot and the foot-walker technology alternatives have greater cost saving potentials than the drone technology alternative. However, depending on future development of the technology, drones can still be a viable solution to assist the delivery of parcels.
Acknowledgments

The author wishes to thank the guidance and patience from Professor Matthew Roorda. I would like to acknowledge the helpful input from Glareh Amirjamshidi, Kianoush Mousavi and Sina Bahrami. I am also grateful to Amelia Ly for always being available for spontaneous thought experiments and theoretical discussions.
Table of Contents

Acknowledgments............................................................................................................. iii
Table of Contents.............................................................................................................. iv
List of Tables .................................................................................................................. vi
List of Figures .................................................................................................................. vii
Chapter 1 Introduction and Literature Review ................................................................. 1
  1.1 Introduction................................................................................................................. 1
  1.2 Literature review......................................................................................................... 4
    1.2.1 VRP with synchronization.................................................................................... 5
    1.2.2 Existing research on movement synchronization............................................... 7
    1.2.3 Metaheuristic solutions....................................................................................... 10
  1.3 Vehicle routing problem with movement synchronization problem description........ 13
    1.3.1 Practical constraints............................................................................................ 14
Chapter 2 Methodology ................................................................................................... 15
  2.1 Mixed integer programming formulation ................................................................... 15
  2.2 Genetic algorithm procedure for VRPMS ................................................................. 19
    2.2.1 Distributed Evolutionary Algorithms in Python (DEAP)................................. 23
    2.2.2 Initialization function......................................................................................... 24
    2.2.3 Split function.................................................................................................... 24
    2.2.4 Evaluation function............................................................................................ 26
    2.2.5 Roulette wheel select function.......................................................................... 28
    2.2.6 Elite select function........................................................................................... 28
    2.2.7 Mate function: partially-mapped crossover...................................................... 28
    2.2.8 Mutate function: inverse indices....................................................................... 30
Chapter 3 Results .............................................................................................................. 32
  3.1 Comparison between exact and metaheuristic methods ............................................ 32
3.2 Empirical results on the three technologies from the metaheuristic method .............34

Chapter 4 Conclusion ..............................................................................................................40

References ..............................................................................................................................42

Appendices ............................................................................................................................46

Appendix A: Detailed summary of performance between MIP and GA .....................46

Appendix B: Detailed summary of route cost reductions .................................................47
List of Tables

Table 1: Functions.................................................................................................................. 23

Table 2: Specification of the heavy and light resource.......................................................... 32
List of Figures

Figure 1: Subtypes of movement synchronization................................................................. 6

Figure 2: Light resource movement pattern (heavy resource is dotted; light resource is solid)... 19

Figure 3: Overview of metaheuristic. .................................................................................... 20

Figure 4: Overview of genetic algorithm.............................................................................. 21

Figure 5: Comparison between MIP and GA solution quality. ............................................ 33

Figure 6: MIP performance over sets by Augerat et al. (1995). ......................................... 34

Figure 7: Route costs between different technologies (standard deviation is shown with black line). .................................................................................................................................................. 37

Figure 8: Capital cost saved from replacing heavy resources with light resources............. 38
Chapter 1
Introduction and Literature Review

1.1 Introduction

At the turn of this century, the concept of e-commerce was ignited by companies such as Amazon, Alibaba, eBay and PayPal. These companies, by capitalizing on the invention of the internet and personal computers, ushered in the age of online transaction of goods. Initially, the bulk of e-commerce sales were business-to-business (B2B) transactions. These transactions are generally delivered with a regular schedule on pallets to business and warehouse districts. Therefore, courier companies could relatively easily pivot its delivery network to handle the increased B2B delivery flows. Then as the barrier to access the internet became lower, business-to-consumer (B2C) transactions increased drastically. It is estimated the total value of global retail e-commerce transactions has grown from 1.34 trillion USD to 2.84 trillion USD from 2014 to 2017 and is projected to increase to 4.88 trillion USD by 2021 (Statista, 2018). The increase of B2C demand has led to more individual parcel deliveries to consumer’s homes at high and irregular frequencies in the delivery network. Individual parcel deliveries mean that the courier company must optimize their capacity dynamically to ensure efficient usage of each delivery resource. The minimization of partial or half-loaded trucks can be an important aspect of a company's operation regime to sustain a profitable business. This is complicated by the irregular delivery schedule created by the consumer's purchasing habits and arrival time expectations. Seasonal trends can vary drastically between holiday shopping months and lull months. This can create a logistic burden on the courier company to scale their warehouse infrastructure in order to meet the peak demand at the risk of inefficient use of space. Lastly, the delivery locations are diluted from concentrated warehouses or business districts to urban, suburban and rural areas where consumers reside. Delivery routes will need to be expanded to cover all the customers if courier companies want to pivot their services from B2B to B2C transactions. In contrast to the B2B delivery flows, these factors make the B2C delivery flows much more difficult for courier companies to incorporate into their existing operations without incurring costly systematic changes.

The shift from B2B to B2C is amplified by the fact that according to the US Department of Transportation (DOT), the value of goods transported by trucks is projected to double by 2040 (United States Department of Transportation, 2017). This forecast could mean that courier
companies will need to be ready to handle an increase in volume while maintaining profitability. It is uncertain whether all this additional value will translate directly to volume of goods. A possible outcome could be that with increased reliability and popularity, online purchase becomes the standard method for purchasing expensive items. Luxury items such as jewelry, expensive electronics, furniture and clothing can still increase in the e-commerce market share. Another trend on the rise is e-shopping for groceries, which are daily or weekly resupplies of basic food items. These resupplies are also heterogeneous and high in volume, often consisting of around 50 items of goods per trip (Kowitt, 2018). Both trends will contribute to the rising value of goods projected by US DOT and add additional complexities for delivery services.

The most critically affected stakeholders are the established courier companies with an existing operation. Companies such as Canada Post, who are responsible for letter mail in Canada, have a vast delivery network reaching to all urban, suburban, and even rural homes. However, their operation was built on a scheduled demand since letter mail did not fluctuate from season to season. Even if there is an increase in volume due to holidays such as Christmas or Thanksgiving, the increase is relatively small and does not add up to create capacity issues for their respective route couriers. Ever since Canada Post has incorporated parcel delivery from e-commerce to their operations, they have experienced new challenges in their letter mail network. The growth of e-commerce and subsequent increase in parcel delivery has put a strain on many other express courier companies to be innovative with their infrastructure and operation. One popular strategy is investing in the option of having the customer directly pick up their purchase at a convenient location in their neighbourhood. Companies such as Canada Post, UPS and Loblaws have introduced variations of direct pick up options for their customers in the form of FlexDelivery, Access Points and Click & Collect, respectively. These facilities are designed to relieve the B2C delivery flows by operating as lockers that securely store the parcel and can be opened by the customer with a unique code. Other logistic strategies include mobile depots which establish a temporary storage hub for smaller delivery units to operate out of in a dense urban area; underground passageways for foot-walkers to deliver to clusters of business buildings; and composite delivery resources.

Of the different strategies, one of the most promising urban logistic solutions from the operation point of view is the composite delivery resource. It is the concept of adding an additional resource to assist the main delivery person operating the delivery truck. This resource can be in the form of
any technology, from the traditional delivery person to the futuristic unmanned drones and delivery robots. Currently, there are various implementations of this delivery operation. Due to the lack of formal regulations and technological stability, the use of drones and delivery robots is rare as opposed to the use of people to assist the delivery operation. In the United States, delivery companies such as UPS and FedEx hire "driver helpers" during peak seasons to assist the driver on their delivery routes. The role of the driver helper is to share the higher work load with the driver by making simultaneous deliveries at each stop. However, they do not drive the vehicle and travel by foot to reach their customers. It is unknown to the author if the driver helper's routes are predetermined using an optimization procedure or if they are coordinated in real-time during the delivery routes. Another variation of this concept is implemented by the delivery companies FedEx and Purolator. FedEx designated their downtown distribution center in Montreal, Canada to facilitate a version of the assisted delivery. Truck drivers meet with on-foot delivery people at this distribution center to split the customers. The truck drivers will ensure deliveries to surface accessible customers while the on-foot delivery people service customers accessible by Montreal's underground pathway system. Similarly, Purolator has a facility located in downtown Toronto, Canada which is used as a distribution center for on-foot delivery. The facility has access to the PATH network which is the underground network in downtown Toronto that connects various office buildings. Both examples demonstrate a strategy which utilizes the existing underground passageway to incorporate aspects of composite delivery resource to their delivery model. Both variations of the composite delivery resource are in need of prior optimization in order to be efficiently operated.

The VRP was formulated to solve the logistics of scheduling a fleet of trucks and was applied by delivery companies among other industries to optimize their daily route scheduling. The advent of the composite delivery resource concept as an urban logistic solution to parcel delivery beckons an extension to the VRP that is suited to schedule both types of delivery resources. This is the motivation for this paper which presents an exact formation of the vehicle routing problem with movement synchronization (VRPMS) and a metaheuristic solution algorithm designed to solve the VRPMS practically for large instances.
1.2 Literature review

The vehicle routing problem has been extensively studied by the research community. Since first introduced by Dantzig and Ramser (1959) as the truck dispatching problem, hundreds of papers from a wide range of journals have proposed solutions to the original and extended variations of the problem. The crux of the problem is to determine the routes for a fleet of vehicles to service a set of customers that minimizes or maximizes the objective function. Usually, the problem is also bounded by a set of constraints which can greatly increase the difficulty to exactly determine the solution. The VRP is usually formulated as an integer programming problem and small sized instances are solved with exact methods such as the branch-and-bound (Balas & Toth, 1983), or branch-cut-and-price (Gauvin, Desaulniers, & Gendreau, 2014). The exact methods are limited to small instances due to its NP-hard nature, thus the computation time for the methods to find the most optimal solution could be intractable. In other words, it is a problem that can be solved with infinite computation power and time, but in practice requires too much computing resources to solve. For large sized problems, heuristic methods have been developed to approximate the solution within a certain threshold of quality from the exact solution but with much faster computation time. Some of the most well-known and commonly used heuristic and metaheuristic algorithms include Clarke and Wright's savings algorithm, sweep algorithm, Christofides-Mingozzi-Toth two-phase algorithm, and tabu search (Laporte, 1991).

Common constraints researchers add to the basic VRP include - but are not limited to - capacity, time windows, multiple vehicle types, pickup and deliveries, backhauls, multiple depots, and dynamic information. These constraints are often mixed and matched to replicate each real-world scenario the researcher is studying and as technology advances more constraints are becoming common. In particular, dynamic information has become more relevant recently due to the advent of communication technology that enables real-time transfer of information between customers, couriers and dispatchers. Literature reviews such as Laporte (1991), Gendreau (2008) and Kumar (2012) summarize the different algorithms, both exact and heuristic, used to tackle these VRP variants. Kumar (2012) noted that a future direction for solving VRP and its variants should involve combining two or more exact methods and heuristics to form metaheuristics.
1.2.1 VRP with synchronization

Of the many variants of VRP, Drexl surveyed the literature and categorized a class of routing problems as the vehicle routing problem with multiple synchronization constraints (Drexl, 2012). Our problem, the VRPMS, is categorized under this class. Normally, vehicles in a VRP are independent of one another, but these problems require an aspect of their operation to be synchronized which leads to an interdependence problem - a change in one route can affect all other routes. Drexl identified five aspects of synchronization: task, operation, movement, load, and resource. Task synchronization occurs when the customer requires one or more suitable vehicles to complete the task. The second synchronization is operation, which ensures the allowable time between the start of a specified operation and the start of another specified operation is satisfied. These operations are performed by suitable vehicles at certain locations. There are three subsets of operation synchronizations: pure spatial synchronization, operation synchronization with precedence, and exact operation synchronizations. Pure spatial synchronization ignores all temporal aspects of the operation, operation with precedence requires a variable offset between the start of each operation, and exact operation specifies the fixed offset between the start of each operation. The distinctive feature of the problem of this paper is its requirement for movement synchronization between the two types of vehicles. This is the third constraint, and it refers to the synchronization of at least two vehicles with respect to space and time at the depot or en route (Figure 1). The vehicle type could be independent which means they can move on their own, or dependent which means they must be pulled by an independent vehicle to move. There are two interrelated decisions with movement synchronization problems, where and when should the vehicles join and separate. Load synchronization means that the capacity of the vehicle needs to be considered. For problems with task vehicles and support vehicles like the trailers and transshipment problem discussed in the next section, the algorithm must account for the load transferred between the two types of vehicles to fulfill the task of the customer. Lastly, resource synchronization deals with the instances where vehicles require the same resource to complete a particular task and the coordination of vehicles such that the limit for the resource is not reached.
The most generic example of this class is the VRP with trailers and transshipment (VRPTT) problem, which contains all the synchronization constraints. The VRPTT problem was originally motivated by milk farms, it requires a heterogeneous fleet of vehicles to transport the milk from the customers to the depot. This fleet includes vehicles of two orthogonal attributes: either independent or dependent, and task (i.e. used to service the customers) and support (i.e. used to carry additional load for the task vehicles). Concretely, the fleet is comprised of lorries (independent task vehicle), tractors (independent support vehicle), drawbar trailers (dependent task vehicle) and semi-trailers (dependent support vehicle). Furthermore, some customers can only be visited by lorries without the trailer. The objective of the problem is to find the best set of routes for the fleet to service the customers and identify the transfer locations between the support vehicles and task vehicles.

Due to the generality of VRPTT, a successful solution algorithm would also provide a means to solve all other problems with fewer synchronization constraints. As noted by Drexel, each synchronization type can be modeled as set of constraints in MIP, and one can theoretically follow the unified framework proposed by Desrosiers et al. (1998) to build this model. However, to solve this MIP model using common practices such as the branch-and-cut-and-price principle, the pricing problem is often difficult to solve without the ability to apply the labelling algorithm. The labelling algorithm is an iterative procedure that gradually improves the label assigned to each node, which represents the cost of the shortest path to that node. This sub-problem has the structure of an elementary shortest path problem with resource constraints, and other approaches besides the labelling algorithm have not been successful (Drexl, 2012). The complexity from the multiple interdependencies of this problem means there is currently no known successful exact solution algorithm for this problem. Moreover, common heuristic solution approaches rely on local search to exploit the property that routes are independent from one another in standard VRPs. However, the interdependence nature of VRPTT dictates that this exploit is no longer available for simple
local search and/or large neighbourhood search-based heuristics. Therefore, even the discovery of heuristic algorithms is a novel research area.

1.2.2 Existing research on movement synchronization

This section reviews the literature specifically on problems featuring elements of movement synchronization en route since that is the closest feature of VRPMS and examine the strengths and shortcomings of the proposed solutions.

Kim et al. (2010) considered a situation where each customer requires several stages of tasks to be completed by specific teams in order. This problem is applicable to the construction of motor ways, oil drills, or other large equipment. However, what differentiates this from normal vehicle routing and scheduling problems is that only the team can perform the task, while the vehicle is used to transport the team to the correct customer. Moreover, each stage of the task is precedent on the completion of the previous stage. Therefore, the fleet of vehicles and set of teams comprised of workers needs to be synchronized such that each team is transported to the correct customer during the appropriate stage. The objective is to minimize the cost of vehicle routing as well as total vehicle distance travelled, total team makespan (time between the start and finish of a sequence of tasks), and total team wait time. A dispatching based heuristic algorithm using the particle swarm optimization technique developed by Kennedy and Eberhart (1995) is used to optimize practically sized instances. It is noted for future work to incorporate other metaheuristic techniques to improve this algorithm (B. I. Kim et al., 2010).

Lin (2008) developed an exact formulation to solve the pick-up and delivery problem with two types of delivery resources (heavy and light) with the ability to travel as a composite unit and to respect time windows. In 2011, he extended the paper by solving real-world instances representing delivery operations in Hong Kong using a two-stage heuristic (Lin, 2011). The utilization of two types of delivery resources is similar to the VRPMS problem, and it is found that enabling coordination of these two resources has benefits over independent operations. However, Lin's formulation contains some assumptions that are relaxed in VRPMS. Firstly, it is assumed that this is an incapacitated problem because it mainly deals with mail. Secondly, there was a maximum limit to the number of light resources each heavy resource could carry. Lastly, the formulation reduced the complexity of the problem by clustering customers with similar pickup times into an aggregate node. The path-based MIP formulation was solved for very small instances, in order to
benchmark against the two-stage heuristic some coordination constraints were relaxed such that instances with 150 customers could be solved. Lin considered two types of coordination between the two resources. The first type is the return transportation, where the light resource is directly dispatched from the depot and returned to the heavy resource en route after servicing a few customers. The second type is the outbound and return transportation, where the light resource is dispatched from a heavy resource en route to service the customers and then returned to the same or different heavy resource. It was assumed that the fixed cost associated with each resource is much greater than the traveling cost. Therefore, the logic of the two-stage heuristic is structured such that the first stage determines the requirements and scheduling of the two types of resources at minimum total cost based on the first type of coordination. A second stage then considers the second type of coordination wherever possible. This leaves the possibility of independent operation of heavy and light resource as feasible solutions of the heuristic. Given these specifications for the heuristic, the simulations showed that as the number of customers increases the coordinated strategy creates more saving by converting heavy resources into cheaper light resources.

More recently, researchers investigated movement synchronization en route, in particular involving drone technology, such as Murray (2015), Di Puglia Pugliese (2017), Ha (2018) and Agatz (2015). In these papers, drones are simplified as a delivery resource capable of making deliveries within a limited service area. It is typical to assume that the drone can make only one delivery (Di Puglia Pugliese & Guerriero, 2017; Murray & Chu, 2015). Murray examined two operational scenarios involving drones for the last-mile delivery. The first scenario arises when the distribution center or depot is located centrally to the customers such that the drones are within range to directly service them from the depot. Murray developed a heuristic solution capable of solving instances with 10 to 20 customers. The second scenario is an extension of the classical traveling salesman problem. Murray and other researchers often refer to it as TSP-D (traveling salesman problem with drone), and it is more flexible and applies to cases when the depot is not centrally located between all the customers. Given the high cost of building depots in urban areas, most express courier companies invest in cheaper depots on the outskirts of the urban core. So, this scenario is potentially more widely applicable to real-world situations. The heuristic developed to solve the TSP-D is based on a route and re-assign framework with the objective of minimizing total service time. Both heuristic and MIP formulations consider a single vehicle and drone,
therefore it is a TSP rather than a VRP solution. Murray also concluded that future work should include mixed operations between the two cases and use simulated annealing, tabu search and other metaheuristic techniques to improve the quality of solution obtained from the heuristics. Ha also tackled a similar variant of this TSP-D and proposed two alternative heuristics which had the objective of minimizing total operation costs. The first algorithm is adapted from Murray (2015) and the second algorithm is based on a split procedure (Ha et al., 2018). From the numerical tests, it was found that by minimizing the cost instead of time a higher utility rate was obtained for the drones. Ha reiterated Murray's point regarding future research direction should include efficient metaheuristics as well as multiple vehicles and multiple drones.

In the same vein, Agatz et al. (2018) also studied the TSP-D and proved the upper-bound on the solution quality of several heuristics they have developed as well as the potential business benefits of this delivery method. Agatz et al. were able to prove the theoretical worst-case approximation bound by developing an integer programming formulation capable of solving 12 nodes exactly. The heuristic approach follows a route-first, cluster-second principle. First, the algorithm solves all the nodes as a standard TSP. Then in the second step, the algorithm splits the truck-only tour to include drone routes. The author introduced two different methods to accomplish the split: a greedy algorithm and an exact partition algorithm. Both methods maintain the order of the customers in the tour as the first algorithm had initially solved it. The last step involves local searches of different initial truck tours from the optimal TSP truck tour for the exact partition algorithm to split. This is done to avoid local minima due to splitting only the optimal TSP solution to obtain the TSP-D solution since there is no evidence that the optimal TSP-D solution should necessarily arise from the optimal TSP solution. The computation time overall, including the local searches, is $O(n^3 \log(n))$ time for the greedy algorithm and $O(n^5)$ time for the exact partition algorithm. Compared to Murray's algorithm which is $O(n^3)$ time for each iteration, the heuristic algorithms in this paper are more computationally expensive. However, the solution quality of exact partition algorithm with full neighbourhood search performed the best amongst these heuristics. Furthermore, with larger instances of beyond 10 customers, which the MIP is limited to solve exactly, the exact partition heuristic consistently outperforms the greedy heuristic despite higher computation times. Agatz et al. (2018) adequately demonstrated the robustness of their heuristic algorithms in solving TSP-D and suggested further research direction towards solving problems with multiple vehicles and drones.
The single vehicle and drone limitation was relaxed to multiple vehicle and drone teams in 2017 (Di Puglia Pugliese & Guerriero, 2017). They formulated this extension with MIP and solved small instances of 5 to 10 customers using CPLEX. The numerical results were heavily dependent on the cost coefficients the authors assigned on the truck and drone respectively, but it did highlight a clear advantage in terms of completion time when drones are included as part of the fleet. It was also confirmed that the route length of the VRP is reduced when drones are included as part of the fleet. This could persuade the adoption of drones to accommodate the increased demand during peak seasons for courier companies. This MIP formulation limited each drone delivery to only one customer, so each drone route consists of a departure leg from the truck at a customer location to its servicing customer and a return leg from the servicing customer to another customer location where it recombines with the truck. The framework presented in this paper is used as the starting point for the MIP model in this paper for the VRPMS.

1.2.3 Metaheuristic solutions

Many of the existing research efforts noted future work should incorporate metaheuristic techniques to further enhance the capabilities of their proposed heuristic solutions, such as Lin (2008), Kim (2010) and Kumar (2012). This section introduces the field of metaheuristics and gives background context to the metaheuristic technique selected for solving VRPMS. Metaheuristics, as defined by Glover (2003), are solution methods that use the interactions between local improvements and high-level strategies to escape local optima and find solutions robustly over the search space. To tackle complex problems such as the VRP and its extensions, metaheuristic methods have emerged as effective tools for practitioners. Compared to an exact solution algorithm, metaheuristics do not guarantee optimality to the problem similar to a heuristic solution algorithm. However, exact solution algorithms oftentimes cannot produce quality solutions for large problems under realistic computation time constraints. Heuristic algorithms, on the other hand, are often problem dependent as they are initially created to solve one specific variation of the VRP. As Gendreau in 2008 has reviewed, many researchers have used metaheuristic techniques to tackle the VRP. The most common metaheuristic techniques used are ant colony optimization, genetic algorithm, simulated annealing, and tabu search. They are used to solve VRP and variations such as time windows, backhauls, pick up and deliveries, multiple use of vehicles, mixed fleet and multiple depots (Gendreau et al., 2008). Brief high-level descriptions of these techniques are presented in the following sections.
1.2.3.1 Ant colony optimization

Ant colony optimization (ACO) is a product of biomimicry. The inspiration for ACO comes from the communication and cooperation mechanisms between ants. In nature, ants leave trails of pheromone to food and water sources and shorter paths are reinforced by stronger pheromone trails because of more ants using that path. As a result, the shortest path emerges from the feedback loop of stronger pheromones attracting more ants which leaves more pheromones. To mimic this phenomenon, the ACO algorithm creates artificial ants to construct greedy solutions randomly every cycle. During the construction process, the algorithm specifies how the ants will evaluate the cost of incorporating the next element and the amount of pheromone it leaves. The pheromone is a metaphorical representation for the memory of the algorithm to determine the quality of each element in a good solution previously constructed by the ants in past cycles.

1.2.3.2 Genetic algorithm

Continuing the theme of biomimicry, the genetic algorithm (GA) metaheuristic mimics the process of genetic evolution and the natural selection process theorized by Charles Darwin in the 19th century. Each potential solution is coded as a type of individual. These are also referred to as chromosomes as a direct analogy to natural selection. Chromosomes are made up of genes which take on different values (alleles) and positions (loci). Starting with a random population of chromosomes, GA evolves the population from one generation to the next by applying operators representing natural processes like selection of the fittest, mutation and mating. Depending on the architecture of GA, different operators are used in combination to simulate this evolutionary process. The goal of GA is to optimize an objective function by mating the good chromosomes of the population from one generation to the next in order to keep the good characteristics, while exploring new characteristics through mating and mutation.

1.2.3.3 Simulated annealing

The real-world analogy for Simulated Annealing (SA) is the physical annealing process which is used to increase the ductility of a material and decrease its defects. It is a randomized local search method looking to improve the system state's energy level with some accepted probability function. Each state corresponds to a solution, and the cost of the solution is its energy. At each
iteration, the algorithm searches for a modified version of the solution with a lower energy level. Then the probability function evaluates the likelihood of accepting the modified solution based on a global time-varying parameter "T" called temperature and the energy of the two solutions. For small values of T, the probability function will favor "downhill" states (i.e. lower energy levels) and avoid "uphill" states. The overall progress of the algorithm is controlled by a predefined schedule, which looks to decrease the temperature parameter over iterations and control the sensitivity of each modification from coarse variations to finer variations. This metaheuristic is an improvement over simple hill climbing or other neighbourhood search methods because it doesn't stop when a solution is found with no better neighbours. The SA method accepts potentially worse neighbours as a means to escape local optima and converge to a global optimum over a long enough time. This results in SA having a relatively slow convergence time for difficult discrete optimization problems (Glover & Kochenberger, 2003).

1.2.3.4 Tabu search

This famous metaheuristic was first proposed by Glover in 1986 in his seminal paper discussing the developments in integer programming and how it has impacted the fields of operations research and artificial intelligence (Glover, 1986). Since then, many papers have recreated the success of using Tabu Search (TS) in solving large combinatorial problems with different constraints. The principle of TS is to search local neighbours for improvements while employing a memory structure to prevent it returning to visited solutions, referred to as cycling. In addition, memory structures can be incorporated to promote searches in promising neighbourhoods or create diversity in the search space. These memory structures are stored as tabu lists. These lists contain the non-improving moves the algorithm has already encountered. In practice, tabu lists only contain limited information to reduce the cost of checking the list each time, such as the last few moves performed on the solution.

These procedures have been used in literature to tackle vehicle routing problems, and various extensions (Gendreau et al., 2008). However, there have not been any applications of metaheuristics to VRP with movement synchronization, since this problem is still relatively novel. Without existing literature, it is difficult to predetermine the best choice for the problem at hand and it is outside the scope of this research to conduct a comprehensive and rigorous comparison between the different metaheuristic procedures. Thus, the selection of genetic algorithm as the
metaheuristic procedure of choice was a matter of the author's preference based on the existing open sourced code framework and the promising features of DEAP (Distributed Evolutionary Algorithms in Python) which is detailed in Section 2.2.1.

1.3 Vehicle routing problem with movement synchronization problem description

VRPMS is an extension of the standard vehicle routing problem by an interdependence feature between the delivery resources. Specifically, there are two elementary types of vehicle in VRPMS - a heavy and a light resource. The heavy resource is an independent delivery vehicle that can move on its own through time and space. The light resource is a special delivery vehicle (or person) that is intended to assist the heavy resource en route by servicing selected customers. The light resource can be transported by the heavy resource and move in time and space on its own. The light resource has a limited range and capacity compared to the heavy resource. On the other hand, depending on the technology, the light resource could have a higher travel speed by virtue of bypassing the congested road network.

The customers can be serviced by either a heavy resource, light resource or both as a unit, and must be visited once and only once. Each time the light resource rejoins the heavy resource, it can be resupplied instantaneously with the demand of following customer(s) provided the demand is within the capacity of the light resource.

Both heavy and light resource must start and end at the depot as one unit. The light resource can depart from the heavy resource at any customer node (launch node) and make deliveries on its own while the heavy resource continues to make deliveries. Then the light resource rejoins the same heavy resource at any future customer node (rendezvous node). The focus on this type of operation for the light resource over direct service from the depot to the customers is motivated by several factors. Firstly, it is conceivable that during the early adoption period of advanced delivery technology for the light resource such as drones and unmanned delivery robots, having human supervision in close proximity is a requirement. So, having the two types of resources servicing the route simultaneously provides this safety net. More critically, the land cost of urban areas is high, and the limited range of the light resource poses a significant challenge for remote depots. Therefore, in order to service customers directly from a depot in the downtown core, the courier company would have to already have a depot in the downtown area or invest significantly to build
one. For example, existing depots used to service downtown Toronto for most delivery companies are located on the outskirts. Given the existing infrastructure of most courier companies it is likely more efficient to adopt a new delivery operation that extends capabilities of each route rather than creating a separate delivery mode using light resources such as drones or delivery robots exclusively.

The objective of the problem is to determine the least expensive set of routes for the heavy and light resources to service all the customers.

1.3.1 Practical constraints

The real-world equivalent to the light resource concept can take various forms. For example, it could be an extra delivery person with a storage capability such as a messenger bag or trolley. Another example could be delivery drones or delivery robots. Each light resource concept imposes a unique set of constraints:

- The number of light resources associated with each heavy resource could:
  - restrict certain launch and rejoin movements; and
  - increase the number of customers the composite team can service during working hours
- The range and capacity of the different light resources due to technology limitations (i.e. battery levels on a drone);
- Physical limitation of delivery person (i.e. max walking distance of a delivery person)
- Customers could limit service to only heavy resources (with or without the light resource); and
- Oversized packages that exceed the carrying capacity of the light resource.
Chapter 2
Methodology

2.1 Mixed integer programming formulation

Parameters:

\( C_1 \) = cost for heavy resource to move one unit length

\( C_2 \) = cost for light resource to move one unit length

\( d_{ij} \) = the distance between customer \( i, j \) for the heavy resource

\( \bar{d}_{ij} \) = the distance between customer \( i, j \) for the light resource

\( t_{ij} \) = time between customer \( i, j \) for the heavy resource

\( \bar{t}_{ij} \) = time between customer \( i, j \) for the light resource

\( s_i \) = time for the heavy resource to service customer \( i \)

\( \bar{s}_i \) = time for the light resource to service customer \( i \)

\( T \) = the maximum time the light resource can wait for the heavy resource

\( E \) = the maximum distance the light resource can continuously travel during one sub-route

\( N \) = set of customer nodes in the instance \([1, n]\)

\( V_L \) = set of nodes the light resource can launch from \([0, n]\)

\( V_R \) = the set of nodes the light resource can return to \([1, n + 1]\)

\( n \) = the number of customers

\( 0 \) = index of the depot

\( n + 1 \) = a copy depot index

\( K \) = set of heavy resources \([1, k]\)
\( D = \) set of light resources \([1, d]\)

\( S = \) set of sub-routes \([1, s]\)

\( v_i - w_i = \) the service time window for customer \( i \)

**Decision variables**

\( x_{ij}^k = 1, \) if resource \( k \) uses arc \((i, j)\) and services customers \( i, j; 0, \) otherwise

\( y_{ij}^{kds} = 1, \) if resource \( d \) of resource \( k \) on sub-route \( s \) uses arc \((i, j)\) and services customers \( i, j; 0, \) otherwise

\( t_{ij}^{kds} = 1, \) if resource \( d \) of resource \( k \) on sub-route \( s \) launches from node \( i \) and services customer \( j; 0, \) otherwise

\( r_{ij}^{kds} = 1, \) if resource \( d \) of resource \( k \) on sub-route \( s \) services customer \( i \) and returns to node \( j; 0, \) otherwise

\( u_i^k = \) integer variable representing the position of customer \( i \) in the route of resource \( k; 0, \) if it is not in the route

\( u_i^{kds} = \) integer variable representing the position of customer \( i \) in the route of resource \( d \) of resource \( k \) in sub-route \( s; 0, \) if it is not in the route

\( a_i^k = \) continuous variable representing the arrival time of resource \( k \) at customer \( i \)

\( a_i^{kd} = \) continuous variable representing the arrival time of resource \( d \) of resource \( k \) at customer \( i \)

**Mixed integer programming model**

Min. \( C_1 \sum_{l \in V_L} \sum_{j \in V_R} \sum_{k \in K} d_{ij} x_{ij}^k + C_2 \sum_{l \in V_L} \sum_{j \in V_R} \sum_{k \in K} \sum_{d \in D} \sum_{s \in S} \tilde{d}_{ij} (y_{ij}^{kds} + l_{ij}^{kds} + r_{ij}^{kds}) \) \( (1) \)

subject to:

\[ \sum_{j \in N} x_{0j}^k - \sum_{l \in N} x_{l,r+1}^k = 0; \forall k \in K \] \( (2) \)
\[ \sum_{i \in V_L} x^k_{ih} - \sum_{j \in V_R} x^k_{hj} = 0; \forall h \in N, \forall k \in K \]  
\[ \sum_{i \in V_L} y^{kds}_{ih} - \sum_{j \in V_R} y^{kds}_{hj} + \sum_{i \in V_L} t^{kds}_{ih} - \sum_{j \in V_R} r^{kds}_{hj} = 0; \forall h \in N, \forall k \in K, \forall d \in D, \forall s \in S \]  
\[ \sum_{j \in N} x^k_{0j} \leq 1; \forall k \in K \]  
\[ \sum_{i \in V_L} \sum_{k \in K} x^k_{ij} + \sum_{i \in V_L} \sum_{k \in K} \sum_{d \in D} \sum_{s \in S} y^{kds}_{ij} + \sum_{i \in V_L} \sum_{k \in K} \sum_{d \in D} \sum_{s \in S} t^{kds}_{ij} = 1; \forall j \in N \]  
\[ \sum_{j \in V_R} \sum_{k \in K} x^k_{ij} + \sum_{j \in V_R} \sum_{k \in K} \sum_{d \in D} \sum_{s \in S} y^{kds}_{ij} + \sum_{j \in V_R} \sum_{k \in K} \sum_{d \in D} \sum_{s \in S} r^{kds}_{ij} = 1; \forall i \in N \]  
\[ \sum_{i \in N} r^{kds}_{ih} \leq \sum_{i \in N} x^k_{ih}; \forall h \in V_R, \forall k \in K, \forall d \in D, \forall s \in S \]  
\[ \sum_{j \in N} l^{kds}_{ij} \leq \sum_{j \in N} x^k_{ij}; \forall h \in V_L, \forall k \in K, \forall d \in D, \forall s \in S \]  
\[ \sum_{j \in V_R} l^{kds}_{0j} = \sum_{i \in V_L} r^{kds}_{in+1} = 0; \forall k \in K, \forall d \in D, \forall s \in S \]  
\[ \sum_{i \in V_L} l^{kds}_{i,n+1} = \sum_{j \in V_R} r^{kds}_{0j} = 0; \forall k \in K, \forall d \in D, \forall s \in S \]  
\[ u^k_i \geq u^k_i + 1 - (n + 1)(1 - x^k_{ij}); \forall k \in K, \forall i \in N, \forall j \in V_R \]  
\[ u^{kds}_j \geq u^{kds}_i + 1 - (n)(1 - y^{kds}_{ij}); \forall k \in K, \forall d \in D, \forall s \in S, \forall i \in N, \forall j \in V_R \]  
\[ u^{kds}_j \geq u^{kds}_i + 1 - (n)(1 - l^{kds}_{ij}); \forall k \in K, \forall d \in D, \forall s \in S, \forall i \in N, \forall j \in V_R \]  
\[ u^{kds}_j \geq u^{kds}_i + 1 - (n)(1 - r^{kds}_{ij}); \forall k \in K, \forall d \in D, \forall s \in S, \forall i \in N, \forall j \in V_R \]  
\[ \sum_{i \in V_L} \sum_{j \in N} l^{kds}_{ij} = \sum_{i \in N} \sum_{j \in V_R} r^{kds}_{ij} \leq 1; \forall k \in K, \forall d \in D, \forall s \in S \]  
\[ \sum_{j \in V_R} y^{kds}_{hj} \leq \sum_{i \in V_L} l^{kds}_{ih}; \forall h \in N, \forall k \in K, \forall d \in D, \forall s \in S \]  
\[ \sum_{i \in V_L} y^{kds}_{ih} \leq \sum_{j \in V_R} r^{kds}_{hj}; \forall h \in N, \forall k \in K, \forall d \in D, \forall s \in S \]  
\[ M(x^{kds}_{ij} - 1) + a^k_i + s_i + t_{ij} \leq a^k_j; \forall k \in K, \forall d \in D, \forall i \in V_L, \forall j \in V_R \]  
\[ M(y^{kds}_{ij} - 1) + a^{kd}_i + \bar{s}_i + \bar{t}_{ij} \leq a^{kd}_j; \forall k \in K, \forall d \in D, \forall s \in S, \forall i \in V_L, \forall j \in V_R \]  
\[ M(l^{kds}_{ij} - 1) + a^k_i + \bar{t}_{ij} \leq a^{kd}_j; \forall k \in K, \forall d \in D, \forall s \in S, \forall i \in V_L, \forall j \in V_R \]
\[
M(r_{ij}^{kds} - 1) + a_i^k + \bar{t}_i + \bar{t}_j \leq a_j^k + s_j; \quad \forall keK, \forall deD, \forall seS, \forall ieV_L, \forall j eV_R \\
M(y_{ij}^{kds} - 1) + a_j^k - \bar{t}_i - a_i^{kds} \leq T; \quad \forall keK, \forall deD, \forall seS, \forall ieV_L, \forall j eV_R \\
M(t_{ij}^{kds} - 1) + a_j^{kds} - \bar{t}_i - a_i^k \leq T; \quad \forall keK, \forall deD, \forall seS, \forall ieV_L, \forall j eV_R \\
M(t_{ij}^{kds} - 1) + a_i^{kds} \leq a_i^k; \quad \forall keK, \forall deD, \forall seS, \forall ieV_L, \forall j eV_R \\
M(r_{ij}^{kds} - 1) + a_j^k \leq a_j^k; \quad \forall keK, \forall deD, \forall seS, \forall ieV_L, \forall j eV_R \\
\]

The objective (1) of the MIP is to minimize the total travel cost for the heavy and light resources. Constraint (2) ensures the conservation of heavy resources. Constraints (3) and (4) are flow balance constraints to make sure that any resource entering a customer node also leaves that node. Constraint (5) limits each heavy resource to only one tour. Constraints (6) and (7) assure each customer is serviced once and only once by either the heavy or light resource. Constraints (8) and (9) ensure light resources are launched and returned to the same heavy resource. Constraints (10)
and (11) restrict the light resource to be directly launched from the depot or directly returned to the depot. In this formulation, the light resources must be carried by the heavy resources in and out of the depot prior to any launch or return movements. Constraints (12) – (15) are sub-tour elimination constraints for the two types of resource. Constraints (16) – (18) ensure the structure of each light resource sub-tour follows the pattern of launch, deliveries and return or launch and return immediately after one customer as illustrated in Figure 2.

![Figure 2: Light resource movement pattern (heavy resource is dotted; light resource is solid).](image)

Constraints (19) – (26) deal with the movement synchronization between the heavy and light resource. Specifically, constraints (19) – (22) assign the arrival time at each customer for both types of resource. The maximum waiting time a light resource is allowed is imposed by constraints (23) – (25). Constraints (26) and (27) guarantee the light resource’s arrival time is before the heavy resource’s arrival time. Constraint (28) ensures the light resource’s maximum travel distance. Constraint (29) represents the capacity constraint for the heavy resource while constraint (30) represents the capacity constraint for the light resource. It is assumed the light resource can instantaneously transfer the load onto the heavy resource upon returning, the capacity constraint for the light resource is restricted to each sub-tour. Constraints (31) and (32) assure the number of light resources launched or returned simultaneously does not exceed the preassigned value. Constraints (33) and (34) assure the number of sub-routes per light resource does not exceed the preassigned value. Constraints (35) and (36) define the time window requirement which the resource can service the customer.

### 2.2 Genetic algorithm procedure for VRPMS

To solve the VRPMS problem, the metaheuristic solves two sub-problems (Figure 3). One of the reasons to split the problem into two sub-problems is that the size of the solution pool for VRPMS is overwhelming even for the GA procedure. The intention of keeping computation time practical...
led to this structure for the metaheuristic. Given a VRPMS problem, the first step is to use GA to obtain the VRP solution with an assumed extended capacity for the heavy resource. This assumption simulates the effects of increased capacity when a composite delivery team of heavy and light resource is used because the composite team can theoretically service more customers within the same operation time. In practice, the true capacity of the heavy resource is usually constrained by the allowable working hours, therefore the heavy resource is often not filled with full capacity. With the additional assistance from the light resource, the heavy resource’s capacity can be extended. Then, each tour in the VRP solution is re-solved using GA as separate TSP problems but with the composite delivery team of both heavy and light resources instead of only heavy resources. The output from this step informs the user how many light resources are required to optimally service each tour in the VRP solution. Then the final solution to the VRPMS is obtained.

![Figure 3: Overview of metaheuristic.](image)

Since both solvers use GA, it is critical to overview the general structure of the GA used. GA can be summarized in five steps:

1. Create individuals to populate the initial generation;
2. Evaluate the fitness of each individual in the population;
3. Select individuals for reproduction;
4. Mate selected individuals to create offspring for the next generation; and
5. Repeat steps 2) - 4) until desired fitness is reached by an individual in the population or the maximum number of generations is reached.

There are many additional operators researchers use to improve the GA process. A popular operator is the one-point mutation, which randomly mutates a single value of the individual to create diversity and prevent the GA from getting stuck at a local optimum. Another common operator is the one-point crossover, which is used to mate the selected individuals by randomly swapping a single value between each other and producing two offspring. The structure of GA used in this paper is illustrated in Figure 4.

![Figure 4: Overview of genetic algorithm.](image-url)
The process shown in Figure 4 is used to solve both the general VRP and the extension with light resources. To begin, an initialization function generates individuals (i.e. potential solutions) to inhabit the first generation of population. Then each individual in the population is evaluated by a function to assess its fitness. The evaluation function works in conjunction with the split function to properly decode each individual into viable solutions to the routing problem. After assessing the fitness of each individual, the best individual is selected as the elite offspring and is preserved into the next generation automatically. At the same time, a pool of potential offspring is selected using the roulette wheel select function. This pool of potential offspring is randomly mutated with the single point mutation function and randomly mated with the partially mapped crossover function. The probability of these evolutionary functions is dependent on the specific parameters used to setup the GA procedure and can be changed accordingly. Finally, a new generation of individuals is created from the combination of evolving the potential offspring and preserving the elite offspring. This new generation is evaluated once again by the evaluation function with the assistance of the split function and if the fitness threshold or the number of generations criteria are met then the GA procedure ends with the best individual in the latest generation as the final output. Otherwise, this process repeats from the selection process in search of better individuals from more generations of evolution.

One of the most challenging aspects of using GA to solve VRP is how the route for each vehicle is coded into the representation of chromosomes which the GA can effectively work with. For a standard VRP with a homogenous fleet, each chromosome can be coded as a list consisting the label of customers as the genes in some order and using a split algorithm to assign sections of the chromosome belongs to each vehicle in the fleet. The split algorithm can be very sophisticated, optimizing how the chromosome is divided, or it can be a simple procedure that splits based on the capacity and range constraints. Then, GA can use this format of chromosome and perform the requisite steps for the evolutionary process. For VRPMS, however, there are two kinds of routes: routes and sub-routes. The routes represent the customers which one set of heavy and light resource will cover. The sub-routes are further classified as either heavy sub-routes or light sub-routes. The heavy sub-route is a subsection of the corresponding master route which only the heavy resource is responsible for, and the similar idea applies to the light sub-route. Each route can be divided into multiple heavy sub-routes and light sub-routes, but more importantly, all customers in the route are also in the sub-routes. In other words, a chromosome which represents a solution to the
VRPMS problem could theoretically be decoded by a split algorithm into a list of lists to represent the routes and sub-routes.

Since GA is built using a variety of functions, some of the functions are written from scratch and others are from open-source projects (Table 1). The two open-source projects used in this paper are DEAP and iROCKBUNNY (2018). DEAP is the engine that powers the genetic algorithm procedure implemented in this paper (Project, 2018). Developed at the University of Laval, DEAP is an open source framework for researchers and programmers to quickly develop prototype applications involving genetic algorithms. The GA procedure was forked from an open source repository by iROCKBUNNY Lab and extended by the author with Python 2.7 and DEAP 1.0 library. The following section presents the high-level summary for each function.

2.2.1 Distributed Evolutionary Algorithms in Python (DEAP)

DEAP is the engine that powers the genetic algorithm procedure implemented in this paper (Project, 2018). Developed at the University of Laval, DEAP is an open source framework for researchers and programmers to quickly develop prototype applications involving genetic algorithms and other evolutionary based techniques. This section will give an overview of the library and the relevant components.

The advantage of using DEAP as the evolutionary algorithm framework is that it allows the user to create types that fit their problem. In other words, DEAP does not assume a list of integers as the representation for the chromosomes in your genetic algorithm. DEAP also gives the user the ability to choose operators to customize the algorithm instead of providing closed and predefined algorithms. The built-in functions from DEAP used in the GA are shown in Table 1.

<table>
<thead>
<tr>
<th>Purpose</th>
<th>Function Name</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create VRP individual</td>
<td>creator.Individual()</td>
<td>DEAP</td>
</tr>
<tr>
<td>Create VRPMS individual</td>
<td>initMVIndividual()</td>
<td>custom</td>
</tr>
<tr>
<td>Create population</td>
<td>initIterate()</td>
<td>DEAP</td>
</tr>
<tr>
<td>Evaluation/Operation</td>
<td>Function</td>
<td>Source</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>----------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>Evaluate VRP individual</td>
<td>evalVRPTW()</td>
<td>iRockBunnyLab</td>
</tr>
<tr>
<td>Evaluate VRPMS individual</td>
<td>evalTSPMS()</td>
<td>custom</td>
</tr>
<tr>
<td>Roulette Select</td>
<td>selRoulette()</td>
<td>DEAP</td>
</tr>
<tr>
<td>Elite Select</td>
<td>selElite()</td>
<td>DEAP</td>
</tr>
<tr>
<td>Mate VRP</td>
<td>cxPartiallyMatched()</td>
<td>custom</td>
</tr>
<tr>
<td>Mate VRPMS</td>
<td>cxSinglePointSwap()</td>
<td>custom</td>
</tr>
<tr>
<td>Mutate</td>
<td>mutInverseIndexes()</td>
<td>iRockBunnyLab</td>
</tr>
</tbody>
</table>

### 2.2.2 Initialization function

The built-in functions included in DEAP are sufficient to create individuals to hold potential solutions to the standard VRP. The individual is a list of integers, which represent the customers. Each individual contains a permutated order of all the customers in the problem. To create individuals to hold potential solutions to the VRPMS, a custom initialization function is coded. The function returns a custom class that is compatible with the DEAP toolbox and can convey the necessary information regarding the heavy and light resource synchronizations. This exemplifies one of the advantages of using DEAP, it is designed to be agnostic to the structure of the evolutionary strategy and the problem.

### 2.2.3 Split function

Since GA encodes the solution as one giant tour, a mechanism is required to decode the giant tour back into distinct tours for each resource. Recall that the chromosomes are essentially lists of customer nodes, GA does not explicitly know where each tour ends or where a new tour begins. Due to this, splitting the giant tour created by GA into a viable VRPMS solution - tours for sets of heavy and light resources - is a critical step. The split procedure concept was first introduced by Beasley (1983) as part of a heuristic for solving VRP. Then Prins proved in (2004) that a metaheuristic can limit the search space to viable TSP solutions instead of VRP solutions because there exists a giant tour which can be split into the optimal VRP solution. There are numerous approaches to creating split algorithms for different VRP variants (Cattaruzza, Absi, Feillet, &
Vidal, 2014; Ha et al., 2018; Prins, 2009; Vidal, Crainic, Gendreau, & Prins, 2013, 2014) and they can range widely in sophistication. The pseudo code for these two split algorithms is listed in Algorithm 1 and 2 respectively.

---

**Algorithm 1**

**Input:** individual, projected capacity (projectedCap)

**Output:** VRP tours (route)

```python
def ind2route(individual, projectedCap):
    route = []
    subroute = []
    for customer in individual:
        capacity += customer.demand
        if capacity <= projectedCap:
            subroute.append(customer)
        else:
            route.append(subroute)
            subroute = []
            capacity = 0
    return route
```

The first split algorithm, which is used to split the giant TSP tour into VRP with only heavy resource, is a procedural capacity and range constraint check. It sequentially holds the customers in the order of the TSP tour. It also keeps track of the cumulative total demand the customers require. Then it splits whenever the demand exceeds the capacity of the heavy resource. At every split, the algorithm restarts the cumulative count of the total demand and distance from zero.

---

**Algorithm 2**

**Input:** heavy resource tour (tour), light capacity (lightCap), light range (lightRange)

**Output:** light resource customers (lightList)
def splitLightCustomers(tour, lightCap, lightRange) is

    lightList = [False] * len(tour)
    distanceList = distanceBetweenCustomers(tsp)
    sortedDistanceList = sort(distanceList)

    foreach i in sortedDistanceList do
        foreach j in customerToEnd(tsp, i) do
            distance = cumulativeDistance(i, j)
            demand = cumulativeDemand(i, j)

            if distance <= lightRange and demand <= lightCap then
                lightList.index(i:j) == [True] * (j – i)
        
    return lightList

The second split algorithm is slightly more complex. It is used to split the VRP tours into VRPMS tours with heavy and light resource sub-routes. Given each heavy resource tour, the algorithm starts a light resource cluster candidate with the closest pair of customers and adds neighbouring customers until the capacity or range constraint is reached. Both split algorithms are key functions as they output the routes in a format such that their costs can be calculated by the evaluation functions described in Section 2.2.4.

2.2.4 Evaluation function

The evaluation function is critical to any GA procedure, as it calculates the fitness of each individual. Fitness is the driving property that advances the evolutionary process, it influences the selection process which in turn affects the mating and mutation processes. In this GA setup, the fitness value is the inverse of the route’s cost, therefore the objective is to maximize the fitness value. The evaluation functions consider the capital cost, unit travel cost, wait cost and delay cost of the heavy resource and the light resource. The capital cost is a one-time cost associated with requiring the resource to be part of the operating fleet. Since there could be differences in how the light and heavy resource navigate the network, different unit travel costs are configured for the resources. Wait cost is associated with a resource waiting for the arrival of another resource before it can proceed with the schedule. Lastly, delay cost is used to represent
the penalty of missing the time windows of the customers. These four cost parameters are input variables set by the user for both types of resources. A scenario analysis of the sensitivities of each of the cost values is presented in Section 3.2 and shows under which circumstances the addition of the light resource makes the operation cost effective.

Algorithm 3

Input: individual with mixed resources (MSindividual), heavy capital cost (initCost), heavy unit travel cost (unitCost), heavy waiting cost (waitCost), heavy delay cost (delayCost), light capital cost (lightInitCost), light unit travel cost (lightUnitCost), light waiting cost (lightWaitCost), light delay cost (lightDelayCost)

Output: fitness

def evaluate(MSindividual, initCost, unitCost, waitCost, delayCost, lightInitCost, lightUnitCost, lightWaitCost, lightDelayCost) is

    lightCost = 0
    heavyCost = 0

    foreach route in MSindividual do

        foreach subRoute in route do

            foreach customer in subRoute do

                calculate distance
                calculate arrivalTime
                calculate elapsedTime

                lightCost = updateCost(lightDistance, lightArrivalTime, lightElapsedTime, lightUnitCost, lightWaitCost, lightDelayCost)

                heavyCost = updateCost(heavyDistance, heavyArrivalTime, heavyElapsedTime, unitCost, waitCost, delayCost)

                routeCost += (lightCost + heavyCost)

            fitness = 1 / routeCost

        return fitness
2.2.5 Roulette wheel select function

Also referred to as the fitness proportionate selection process, this function is used to select potential solutions in the GA procedure. Given the fitness of an individual \( i \) is \( f_i \), then the probability of being selected by this function is \( p_i = \frac{f_i}{\sum_{j=1}^{N} f_j} \) where \( N \) is the total number of individuals in the population. To visualize the process, imagine a roulette wheel and each slice of the wheel represents an individual's fitness. If the individual has a higher fitness, it has a larger slice on the roulette wheel and each selection is a spin of the roulette wheel. This selection process gives the chance for weaker solution to be selected; this is advantageous because it potentially keeps useful characteristics of weaker solutions in the population for future generations.

2.2.6 Elite select function

This built-in function ranks the fitness of the individuals and selects a specified number of individuals from the best to worst. This is less sophisticated than the roulette wheel select function but guarantees the best individuals of each generation are never lost. In the animal breeding industry, this is a standard method known as truncation selection.

In this GA procedure, only one single elite is selected at each generation and the rest of the population is selected using the roulette wheel method.

2.2.7 Mate function: partially-mapped crossover

When using GA to solve VRP, the ordering of the genes in the chromosome becomes very important. This is different than other popular applications of GA where the values of the genes are the sole factor to the solution. As noted by Goldberg (1985) in his seminal paper introducing a new type of crossover operator known as the partially-mapped crossover (PMX), normal GA procedure assumes that the values (alleles) of the chromosomes decode independent of their position (loci). Given how the genes of VRPMS chromosome represent the order in which the heavy and light resource service the customers, the loci of the genes will directly affect the fitness of the chromosome. Prior to the introduction of PMX, none of the available crossover techniques preserved the qualities of value and ordering simultaneously in the chromosomes. Thus, ordering problems such as the TSP and VRP were difficult and deemed GA-Hard (Goldberg & Lingle, 1985). PMX can exchange ordering and value information between two chromosomes and is suited
to solve problems like VRPMS. The mechanics of PMX are as follows. First a mapping section, where the mating procedure will take place, is defined between two random positions along the string for both chromosomes. Then each value in the mapping section is sequentially swapped in the mapping operation between the two chromosomes. During each swap, if the newly swapped value is a duplicate of another existing value at another position, then the existing value takes on the swapped out value. To illustrate this, consider the following two strings of A and B.

A = 1 2 3 4 5 6 7 8 9
B = 5 3 1 2 7 4 8 6 9

If the random mapping section is set to positions 4 and 7, then the two mapping sections are: 4-5-6-7 for string A, and 2-7-4-8 for string B. The mapping operation of B onto A follows the steps of swapping the 4 and the 2, the 5 and the 7, the 6 and the 4, and the 7 and the 8, or whatever new values those positions take on during the mapping operation. In the first mapping operation, swapping the 4 and the 2, string A has a duplicate value 2 in position 2 so it will be replaced by swapped out value of 4. The same procedure continues for the next three mapping operation resulting in strings A' and B'.

A' = 1 4 3 6 7 2 8 5 9
B' = 7 3 1 4 8 6 5 2 9

The offspring A' and B', from visual inspection, retain some of the propagated ordering from their parents A and B. It is also proved by Goldberg in his 1985 paper, that PMX can explore both allele and ordering combinations in parallel implicitly. This function is built into the DEAP library but was slightly modified to work with the structures of the individuals specified.

The function to mate VRPMS individuals together is simpler than PMX because less exploration is required. It mimics a single point swap crossover, which is a simple mating method for two individuals by switching the value at one random location. To visualize this, reconsider strings A and B.

A = 6 [1 2 3 4 5] 7
B = 8 [1 2 3 4 5] 9
By inspection, these two strings are slightly different as they contain a nested list. These is a standard representation of a VRPMS individual, which the nested list corresponds to the route of the light resource and the values bordering the nested list represent the launch and rendezvous nodes. To execute the single point swap crossover, the launch and/or rendezvous nodes are exchanged between A and B to form new strings A’ and B’. In this scenario, only the launch node is exchanged.

\[ A' = 8 \ [1 \ 2 \ 3 \ 4 \ 5] \ 7 \]

\[ B' = 6 \ [1 \ 2 \ 3 \ 4 \ 5] \ 9 \]

The custom mate function for VRPMS individual is programmed to search potential launch and return locations for the light resource. This is accomplished by swapping the possible rendezvous points along the heavy resource’s route between individuals. Since the order of the customers is determined a priori, the mating process in this step is intended to search local optimizations with launch and return points for the movement synchronization.

2.2.8 Mutate function: inverse indices

In a simple example where the chromosomes of a GA are comprised of binary values of 0 or 1, a viable mutation operator is to select a value randomly along the chromosome and flip the value. This is known as a bit string mutation and is visualized by the following example showing string A undergoing the mutation to become string B.

\[ A = 0 \ 0 \ 0 \ 0 \ 1 \ 0 \]

\[ B = 0 \ 0 \ 0 \ 0 \ 0 \ 0 \]

Since the individuals in this GA process are non-binary values, mutation operators such as bit string mutation or flip bit are not applicable. Instead, the mutation operator implemented selects a random segment of the chromosome and inverses the order. To illustrate the effect, consider string A and a segment defined by randomly generated indices 1 and 3.

\[ A = 4 \ 5 \ 1 \ 2 \ 8 \ 3 \ 6 \ 7 \]
The lower value is the starting position of the segment and the higher value is the ending position. Thus, the segment defined by indices 1 and 3 is [4 5 1]. The function takes the segment and produces a corresponding segment with inversed indices [1 5 4] and inserts into string A to produce A’.

A’ = 1 5 4 2 8 3 6 7

The purpose of a mutation operator in the overall GA process is to introduce genetic diversity and prevent the algorithm from getting stuck in local minima. The probability an individual undergoes mutation should be determined by a user-defined probability. This probability value should be relatively low, otherwise the GA process becomes a random search since no genetic features are preserved.
Chapter 3
Results

This section applies the methodology in Chapter 2 to identify the desirable characteristics of the technology that could enable the best use of light resources for the last-mile delivery service. In addition, the solution quality of the metaheuristic algorithm is compared against the exact formulation.

The hardware setup used to obtain the empirical results is comprised of a 2.5 GHz Intel Core i5 CPU with 4GB 1600 MHz DDR3 of memory. The MIP model was coded with Python 2.7 and Gurobi 7.5.2. library. It was configured to timeout at 9999 seconds or stop at a tolerance of 1.00e-04. Fixed configurations for the GA are: mating ratio (90%), mutating ratio (5%) and population size (20n), where n is the number of customers in the problem. Other configurations for the GA varied depending on the problem to minimize the runtime while maintaining performance.

3.1 Comparison between exact and metaheuristic methods

A library of 30 instances was solved with the exact MIP model and metaheuristic GA procedure. The instances are randomly generated to evaluate the difference in performance between the exact formulation and the metaheuristic algorithm. Each instance contains either 10, 15 or 20 customers distributed over a 100 by 100 unit length square region. Each customer requires a demand of either 2 pieces or 5 pieces and a service time of either 10 unit time for the heavy resource or 5 unit time for the light resource. The depot is located either at the origin or the center of the square region. The heavy and light resource are specified with certain characteristics, which are summarized in Table 2. The difference in capacity and range is to simulate the average characteristics of the light resource technologies. The travel cost per unit length is the same between two resource types while the travel speed is faster for the light resource. This is done to encourage both methods to consider light resource cooperation in the final routing solution.

Table 2: Specification of the heavy and light resource.

<table>
<thead>
<tr>
<th>Resource Type</th>
<th>Capacity [piece]</th>
<th>Range [unit length]</th>
<th>Travel cost per unit length</th>
<th>Travel Speed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Heavy</td>
<td>30</td>
<td>unlimited</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
As shown in Figure 5, the solution cost gap for the VRPMS is not as tight as the TSP. The average solution gaps for VRPMS are 31%, 32% and 27% for the 10, 15 and 20 customers size instances respectively. This is because of the rudimentary split method employed to enable the GA procedure to solve VRPMS. This simple approach to sequentially group the light customers with a greedy algorithm may not consistently produce good solutions for the GA to iterate on. As a result, the average solution gap is between 20% to 40%. The solution gap slightly improves with 20 customers, but this could be due to the fact that the MIP model is reaching the stipulated computation time limit and not fully converging. This is an area that future work could improve on and is further discussed in Chapter 3. Despite the average solution quality, it does not diminish the potential GA has for solving combinatorics problems because it is shown to be very efficient at searching for the TSP solution with the average solution gap to be below 20% and on par with the MIP model for the 10 customer instances. Additionally, the GA procedure’s computation time is drastically faster the MIP model. The average runtime for the MIP model to solve the 10 customer instances is 4,639 seconds, compared to 22.9 seconds for the GA procedure. This also allows for more sophisticated split algorithms to be employed, as they most likely will incur higher computation costs.

![Figure 5: Comparison between MIP and GA solution quality.](image)
The MIP model was also tested against known VRP instances and solutions from literature. From Set A, B and P by Augerat et al (1995), those with less than 40 customers were used to validate the MIP model at solving regular vehicle routing problems. The performance of the MIP is summarized in Figure 6. Eight instances with less than 30 customers were selected from Set P, while 16 instances were selected from Set A and B which contained customers ranging from 30 to 40. The larger instances proved to be computationally expensive for the MIP model to solve. This resulted in almost all the runs from Set A and B to reach the computation time limit of 9,999 seconds with high solution bounds, thus creating the larger variance in performance compared to Set P. The average solution gap for Set P is 0.97% which indicates the MIP model is producing nearly identical solutions to literature established ones. For Set A and B, the solution gaps are 9.40% and 3.75% respectively. Moreover, there are more outliers in Set A as indicated by the larger variance of the box graph.

![Figure 6: MIP performance over sets by Augerat et al. (1995).](image)

### 3.2 Empirical results on the three technologies from the metaheuristic method

Currently, there are no established financial numbers on the different types of technology that could operate as the light resource. Initial pilot tests and economic estimates indicate the cost per stop is cheaper for technologies such as drones compared to traditional delivery vans (E. Kim,
For the last-mile component of the delivery, a USPS van averages $2.50 for a typical shoebox delivery and drone/robot technology could cost around $1.74 per delivery according to a report published by Skylar Drone Research (Jenkins, Vasigh, Oster, & Larsen, 2017). This is estimated from a combination of capital cost, operation and maintenance cost, motor cost, battery cost, and electricity cost assuming 50 hours of work each week. As the technology continues to mature, the average cost per stop should continue to decrease. The last advantage of the drone is its ability to fly over traffic congestion and road network completely. Theoretically this enables drones to significantly reduce the travel time compared to the delivery van to service the customers.

Sidewalk robots are at an even earlier stage of development. Very few companies are investing in the concept of building a delivery solution with autonomous vehicles capable of navigating the sidewalk to make deliveries. This technology also faces legislative roadblocks for commercial use and does not have the same consumer uses as a drone (Simon, 2017). Thus, capital cost estimates for this technology range widely from $2,000 to $7,000 to purchase a robot depending on how the companies scale if the delivery option becomes mainstream (Huang, 2017; Pettitt, 2015). The projected cost per stop for the sidewalk robot ranges from low to almost negligible, like the drone, but the sidewalk robot can carry much higher loads per trip since it moves on the ground. This leads to the slow travel speeds these robots tend to operate under since they need to blend with pedestrian traffic without causing disruptions. People move more erratically than vehicles since there are no clear lanes or traffic rules, so the sidewalk robot could require a high degree of caution while operating that leads to slower travel speed.

The most established variation for a light resource is the foot walker, which is an extra delivery person with a cart or bag. Foot walkers are widely used by courier companies during peak seasons to help manage the high demand to ensure the routes are completed within the regular work hours. The advantages of using the foot walker are reliability, range and capacity. There are many subtle decisions underpinning each delivery and often they are completed seamlessly by a human but can create unexpected difficulties for automated technology. Until automated technology is fully adopted and regularly used by courier companies, the initial adoption phase could keep the promised low delivery cost per stop relatively high. The foot walker also does not have the same limitation with its range in most use cases as the battery-operated technologies do. With a handcart or bag, the capacity of the foot walker is significantly higher than a drone and comparable to a technologically matured sidewalk robot. The disadvantage is mainly the cost. To have seasonal
workers to accommodate demand requires a hiring process, training sessions and appropriate compensation. This accumulates to a higher capital cost and travel cost comparable to automated technology. Another minor disadvantage is travel speed, it is most likely faster than the sidewalk robot but considerably slower than the drone.

The light resource is not penalized as much for waiting for the heavy resource at the return location. Whereas the heavy resource is penalized for waiting for the light resource at a ratio of 5 to 1. This is to disincentivize the algorithm to search solutions where the heavy resource is in idle. Generally, the operation cost is higher for the heavy resource, since the delivery person is paid an hourly wage and the drone is not.

Assumptions about the capital cost, delivery cost per stop, capacity and travel speed for the heavy resource and various versions of the light resource technologies are detailed in Table 3. These figures represent the typical cost for each light resource technology. It is observed that each technology has its own distinct advantages. The drone has the lowest capital cost and a relatively low travel cost, the sidewalk robot has the lowest travel cost and the walker has the highest capacity and biggest range. Special consideration is made for the capacity of the heavy resource. It must be small enough to enable multiple routes as potential solutions to the instances. Since the instances range from 50 to 100 customers with demand of 1 piece, an extended capacity of 100 pieces is assumed as the base case for the delivery van. The other parameters of the heavy resource are assumed to be $80,000 of capital cost, $0.10 per unit length of travel cost, travel speed of 1 unit length/unit time, wait cost of $0.05 per unit time and delay cost of $0.01 per unit time.

**Table 3: Technology cost ratios with respect to the heavy resource.**

<table>
<thead>
<tr>
<th>Technology</th>
<th>Capital cost ratio</th>
<th>Capacity ratio</th>
<th>Range of L [unit length]</th>
<th>Travel cost ratio [$/unit length]</th>
<th>Travel Speed</th>
<th>Wait Cost [$/unit time]</th>
<th>Delay Cost [$/unit time]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Drone</td>
<td>25:1</td>
<td>100:1</td>
<td>200</td>
<td>5:1</td>
<td>1:5</td>
<td>1:0</td>
<td>1:1</td>
</tr>
<tr>
<td>Robot</td>
<td>15:1</td>
<td>25:1</td>
<td>500</td>
<td>10:1</td>
<td>1:1</td>
<td>1:0</td>
<td>1:1</td>
</tr>
</tbody>
</table>
The sensitivity analysis of each light resource technology is performed over an instance set with 50 and 100 customers and technology parameters specified according to Table 3. The instances are generated following the same properties as Section 3.1 except the size of the instance is a square of 500 by 500 unit length and every customer has a fixed demand of 1 piece. Each instance is solved 10 times with a randomized starting population for the genetic algorithm. The number of generations used in the GA procedure varied from 300 to 500 depending on the size of the problem and how long it took to converge.

The solution results are illustrated in Figure 7. The biggest takeaway is drone technology did not significantly decrease the overall route cost. This is mostly likely because each drone can only make one delivery at a time within a small range, so the number of potential customers for drone delivery is low. From this analysis, if the capacity of the drone does not increase, drone technology is not a competitive alternative. With a moderately sized capacity and range and lower travel cost, the sidewalk robot alternative performed better with a 10% decrease in travel cost despite its slower speed parameter. Lastly, with a high capacity and range, the foot-walker alternative produced the largest decrease in travel cost with a 16% reduction, despite its higher unit travel cost and waiting cost.

<table>
<thead>
<tr>
<th>Walker</th>
<th>8:1</th>
<th>10:1</th>
<th>1000</th>
<th>5:3</th>
<th>1:2</th>
<th>5:2</th>
<th>1:1</th>
</tr>
</thead>
</table>

Figure 7: Route costs between different technologies (standard deviation is shown with black line).
This leads to an interesting discussion regarding the relationship between the capital costs of each technology. In practice, the underlying reason for using multiple delivery vans to service an area is usually not because of capacity issues but limitations on the length of a work shift. To replicate this scenario, the GA procedure is configured with a relatively tight end service time across all customers. This requires the solution to contain more than the mandatory amount of heavy resources as defined by the capacity constraint alone. For example, in the scenario with 50 customers, it is technically possible to service all 50 customers with one heavy resource based on the capacity constraint. However, the GA solution often employs two heavy resources and each heavy resource utilizes approximately 25% of its full capacity. Therefore, through the addition of light resources, each heavy resource’s route can theoretically increase in size and potentially eliminate some of the routes. From the GA solutions for the 50 customer and 100 customer instances, the average utilization rate is calculated for each resource type. Next, given the capital cost ratios between the heavy resource and each light resource technology, the relative savings from capital costs from different customer sizes are calculated and summarized in Figure 8.

![Figure 8: Capital cost saved from replacing heavy resources with light resources.](image)

The general trend illustrated in Figure 8 shows that as the number of customer increases there is more potential to save on capital costs. This makes intuitive sense because larger instances require more heavy resources, so adding the light resource cooperation can eliminate more heavy
resources which creates larger savings. The limited capacity of the drone means that many drones are required to replace the capacity of one heavy resource, and the relatively lower capital cost does not offset the total number of drones required. As shown in Figure 8, the drone alternative consistently produces the smallest capital cost savings. Meanwhile, both robot and walker technologies produce larger capital cost savings. Compared to the foot-walker, the robot technology is more favourable for smaller customer sizes. Once the instance size passes a critical value between 100 and 200 customers, the foot-walker technology is the cheaper option. Recall that the foot-walker alternative also incurs the highest route cost savings, thus making it the most favourable option out of the three technology alternatives.

It is notable that the ratio between the number of walkers and heavy resource is limited to 2:1. In other words, each heavy resource can carry at most two walkers as an effort to keep the scenario realistic. Following the same logic, the number of robots per heavy vehicle is limited to two and the number of drones per heavy resource is limited to three.

The comparison analysis demonstrates that the relationship between the parameters of heavy resource and light resource needs to be closer to the foot-walker technology alternative in order to be cost effective. This means that for drone technology to supplant the traditional delivery person, the capacity needs to be increased or the capital and operating costs are further reduced than the assumed values in this analysis. As technology continues to mature and advance, it is likely that manufacturers will achieve better parameters for drones and sidewalk robots than the conservative assumptions made in this paper.
Chapter 4
Conclusion

The results presented in this thesis contribute to the extensive literature in operations research by introducing a model for cooperative delivery schemes through movement synchronization between delivery resources. This is a relatively novel topic within operations research with limited published work. It is also a new area in practice, the unmanned vehicle technology described such as drones or sidewalk robots are in early stages of development. Only recently, delivery companies such as Amazon have announced their plans to use drones in their delivery infrastructure. It is uncertain how the various roadblocks such as policy or public acceptance will influence the eventual operation of these light resources. Nevertheless, this is a direction which courier companies have identified as a potential solution to urban logistics challenges which are exacerbated with the shift in consumer’s shopping habits to e-commerce. This paper provides a method to optimize this new mode of delivery and assess the cost implications of incorporating new technologies. A mixed integer programming model is formulated to provide a mathematical description of VRPMS and to exactly solve small instances with less than 20 customers. The metaheuristic model is developed to solve larger instances with the intent of quantifying the trade-offs between different light resource technologies. By using the genetic algorithm as the main tool, the metaheuristic model is capable of solving TSP relatively robustly and showed promise in solving VRPMS.

The insights gained from the cost analysis produced by the metaheuristic is valuable for express courier companies to rationalize committing research and development efforts into integrating promising light resource technology such as drones or sidewalk robots. Out of the three light resource technology alternatives, drone technology did not perform as well as the others in this cost analysis. This should not deter the progress drone manufacturing companies are committed to make. However, it should serve as a reminder that in order for the drone to be a viable delivery resource for express courier companies, manufacturing companies either have to increase the carrying capacity and range or drastically decrease the capital cost and operation cost from the assumed values.

Further research should focus on improving the greedy clustering component of the metaheuristic algorithm. The best routing sequence with both heavy and light resources does not necessarily
stem from the best routing sequence with only the heavy resource. This further research to find more optimal split algorithms is challenging but is considered to be outside the scope of this paper. Second, the time window constraint considered in this paper is limited to a static range. A better reflection of reality could be a dynamic time window for each customer and each technology depending on the mode of reception the customer has indicated. Third, drone and sidewalk robot technologies are rapidly changing, causing their attributes to be uncertain. The cost of delivery truck and foot-walker operations also vary between companies. So consistent and timely updates of the attribute assumptions made in this paper are required in any application. Also, the customer locations are randomly generated and does not correspond to actual delivery areas of courier companies. This could be improved by creating instances with real customer locations in downtown or rural areas. Then the cost analysis could extend to different scenarios involving demographic changes. Lastly, as discussed in Section 1.2.3 there are other metaheuristic algorithms available to solving routing problems. Other techniques may have the potential for computation gains and better solution bounds for this metaheuristic. For the cost-benefit study of various light resource technology, the procedure developed was sufficient. However, for a company to fully rely on the routing algorithm to solve day-to-day operations with the heavy and light resource, a tighter bounded algorithm would be ideal.
References


43


Appendices

Appendix A: Detailed summary of performance between MIP and GA

<table>
<thead>
<tr>
<th>Customer Size</th>
<th>Solution Method</th>
<th>Cost</th>
<th>Runtime</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Avg</td>
<td>Min</td>
</tr>
<tr>
<td>10</td>
<td>MIP-TSP</td>
<td>1,933</td>
<td>1,415</td>
</tr>
<tr>
<td></td>
<td>GA-TSP</td>
<td>1,954</td>
<td>1,470</td>
</tr>
<tr>
<td></td>
<td>MIP-VRPMS</td>
<td>1,531</td>
<td>868</td>
</tr>
<tr>
<td></td>
<td>GA-VRPMS</td>
<td>1,963</td>
<td>1,482</td>
</tr>
<tr>
<td>15</td>
<td>MIP-TSP</td>
<td>2,440</td>
<td>1,658</td>
</tr>
<tr>
<td></td>
<td>GA-TSP</td>
<td>2,613</td>
<td>1,852</td>
</tr>
<tr>
<td></td>
<td>MIP-VRPMS</td>
<td>1,997</td>
<td>1,181</td>
</tr>
<tr>
<td></td>
<td>GA-VRPMS</td>
<td>2,584</td>
<td>1,578</td>
</tr>
<tr>
<td>20</td>
<td>MIP-TSP</td>
<td>2,903</td>
<td>2,201</td>
</tr>
<tr>
<td></td>
<td>GA-TSP</td>
<td>3,349</td>
<td>2,359</td>
</tr>
<tr>
<td></td>
<td>MIP-VRPMS</td>
<td>2,820</td>
<td>1,500</td>
</tr>
<tr>
<td></td>
<td>GA-VRPMS</td>
<td>3,536</td>
<td>2,379</td>
</tr>
</tbody>
</table>
## Appendix B: Detailed summary of route cost reductions

<table>
<thead>
<tr>
<th>Technology</th>
<th>Customer Size</th>
<th>Average route cost</th>
<th>Cost reduced</th>
</tr>
</thead>
<tbody>
<tr>
<td>Drone</td>
<td>50</td>
<td>3,355</td>
<td>-0.65%</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>6,596</td>
<td>-1.09%</td>
</tr>
<tr>
<td>Robot</td>
<td>50</td>
<td>3,020</td>
<td>-10.56%</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>5,852</td>
<td>-12.24%</td>
</tr>
<tr>
<td>Walker</td>
<td>50</td>
<td>2,829</td>
<td>-16.22%</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>5,460</td>
<td>-18.12%</td>
</tr>
</tbody>
</table>