Modelling and Solving the Senior Transportation Problem

by

Chang Liu

A thesis submitted in conformity with the requirements
for the degree of Master of Applied Science
Graduate Department of Mechanical and Industrial Engineering
University of Toronto

Abstract

Modelling and Solving the Senior Transportation Problem

Chang Liu
Master of Applied Science
Graduate Department of Mechanical and Industrial Engineering
University of Toronto
2018

As the Canadian population ages, there is an increasing demand for senior transportation services that cannot be met by regular public transportation systems. This thesis aims to bring attention to the combinatorial optimization problem of senior transportation. By extending current vehicle routing problem knowledge and computational technologies, we construct novel problem definitions for the Senior Transportation Problem (STP) and the Senior Transportation Problem with Overbooking (STPOB) based on data provided by a partnering non-profit organization specializing in senior transportation services. Solution approaches including mixed integer programming, constraint programming, two logic-based Benders decompositions and a construction heuristic are developed and empirical analyses on both randomly generated datasets and real-life datasets are performed. Constraint programming demonstrates best results being able to solve to optimality large real-life instances of up to 270 vehicles with 385 requests for the STP and up to 210 vehicles with 320 requests for the STPOB under 600 seconds.

Acknowledgements

This thesis would not have been possible without a number of people. I would like to use this space to express my gratitude towards them.

Before all, I would like thank my supervisors Professor J. Christopher Beck and Professor Dionne M. Aleman for the boundless guidance, patience, support, as well as all the creative ideas throughout the past few years. Their persistence has inspired and pushed me to places and opportunities that I could not even have imagined previously. Not only have they advised me on technical expertise, they have shown me how interesting and fun research can be. I could not have imagined better mentors for my studies and research.

I would like to thank my committee members, Professor Merve Bodur and Professor Chi-Guhn Lee for their time and feedback.

Thanks to all the amazing people at TIDEL and morLAB for such a fun and amazing time that I would never forget in my life. They have made the sometimes dull student life shine in bright colours.

Thanks to my parents for the unconditional love and support for me to pursue my dreams.

Lastly, to Chong, my best friend and beloved husband, thank you for being always by my side to share the bitterness, keep me sane and make me laugh.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The elderly population has been increasing rapidly worldwide. In a report from the United Nations published in 2015, the percentage of the population aged 60 years or older was 12.3% in 2015, but is expected to grow to 16.5% in 2030 and 21.5% in 2050 [51]. With this large increase in elderly population, communities need to make sure that the infrastructures are in place to support the everyday life of this population. One major component is transportation; whether it is from home to a hospital or simply a grocery run, transportation plays a crucial role in every elder's daily activities. Currently, there are some non-profit organizations that provide these "senior transportation" services in many communities. In the Greater Toronto Area, for example, organizations such as Toronto Ride, Circle of Care, and Community and Home Assistance to Seniors (CHATS) provide services to all eligible elders in their districts. However, the resources available for these transportation services are often very limited and many elders are left on the waiting list for a long time. Furthermore, due to lack of scheduling experts, the schedules assigned to each driver are often sub-optimal in a sense that many vehicles do not operate at full capacity. Therefore, finding optimal schedules is crucial for organizations to reduce operational costs and meet necessary demands.

This thesis is concerned with formally defining the Senior Transportation Problem (STP) and the Senior Transportation Problem with Overbooking (STPOB) and presenting different approaches to solve both of these problems. Five different methodologies including Mixed Integer Programming (MIP), Constraint Programming (CP), Logic-based Benders Decomposition (LBBD), and Heuristics are explored to tackle each of the STP and the STPOB and empirical analysis has been performed in order to understand the reasons why one approach performs better than the other approaches.

1.0.1 Characteristics of the Problems

The STP is a transportation problem and draws many of its problem characteristics from different variations of the Vehicle Routing Problems (VRP). The VRP is a group of combinatorial optimization problems that have been extensively studied in the literature as they have demonstrated to be significant in today's business operations [63]. Given a set of transportation requests and a fleet of vehicles, the VRP aims to find a set of routes that can satisfy all requests while obeying all routing constraints at a minimal cost (or using the least number of vehicles). Some variations of the VRP include adding capacities to vehicles, time window constraints to transportation requests, sequence constraints on request locations, and stochasticity on the demand of requests. A thorough review of the VRP is provided in Chapter 2.

To understand the nature of the STP, we have partnered with CHATS which provides assistance, including transportation services, to seniors in the York and South Simcoe regions in Ontario, Canada. In 2015, CHATS provided 88,000 trips to more than 7,300 seniors. After multiple conversations with CHATS, we have learned that, unlike traditional transportation problems that minimize the number of vehicles used and/or the total distance travelled, CHATS' objective is to maximize the number of clients served. There are two types of drivers: paid drivers and volunteer drivers who have vehicles of different capacities and also depart from different locations. In addition to routing and time window constraints, additional constraint such as maximum ride time of a client must also be enforced since the clients are seniors and user comfort is also very important.

Another aspect of the problem faced by CHATS is that requests are frequently cancelled. Often, due to weather conditions or illness, from 15% to 26% of the requests are cancelled in the last 24 hours. Therefore, the second question that we would like to research is how to create the most cost effective schedule given the additional information on each request's probability of cancellation. The approach that we consider is to allow overbooking: any request that is cancelled does not incur any cost and overbooked requests are served by a higher cost taxi.

1.0.2 Thesis Outline

Chapter 2 provides a literature review on various transportation problems and the different methodologies to solve these optimization problems. This chapter concludes with a discussion of overbooking.

The thesis follows with Chapter 3 which explains the process of generating test problem instances to assess the performance of the different algorithms. Two sets of dataset are employed: randomly generated problem instances and real-life problem instances extracted from the CHATS' database. Additional data analyses on the CHATS dataset are also described in this chapter including the use of prediction models to compute the probability of cancellation of a given request.

In Chapter 4, a formal problem definition for the Senior Transportation Problem is introduced along with the parameters and notation used in the rest of this thesis. Five approaches including the variable definitions and formulations or pseudo codes are presented next. All algorithms are tested on the generated datasets and detailed analysis on the CP model and the LBBD approaches is performed to understand the behaviour of these algorithms. The best performing approaches are then tested on the CHATS datasets to assess the different algorithms on real-life instances.

Chapter 5 follows with the introduction of STPOB. As in Chapter 4, five approaches and their formulations are presented and analysis on the different algorithms is presented. Additionally, Chapter 5 discusses the difficulty of the STPOB compared to the STP. Finally, the best performing approaches are again tested on the CHATS instances.

The thesis concludes with Chapter 6 which presents a summary and conclusions drawn from this research followed by a discussion of directions for future research.

Chapter 2

Literature Review

The transportation of goods and people is an important part of everyday life. In order to increase the efficiency of the transportation industry and economize on resources, vehicle routing problems (VRP) have been widely studied in the field of Operations Research. Laporte highlights the research discoveries and the development of highly sophisticated exact algorithms and heuristic methods of VRPs in the past fifty years [41]. VRP can be defined as the class of combinatorial problems to find a schedule for vehicles to visit a set of locations while optimizing some aspects of transportation costs. This class of problems is very complex and intricate; the problem and its various variants have been proven to be NP-hard [45, 28] and require a lot of computational power to be solved to optimality. Furthermore, many complicated constraints arise from real-world problem examples. In order to address the difficulties of transportation problems, many variations of VRP have been studied in the literature. Some of the most researched VRP variants include time windows on visits to locations (VRPTW) [36], pickup and delivery requirements (PDP) [20] and vehicle capacities (CVRP) [55].

The research on VRP is still very active but has shown a shift to more realistic problems that involve a variety of constraints that represent the real-world. In 2014, Caceres-Cruz et al. surveyed existing problem definitions and the corresponding state-of-the-art solution techniques [10], emphasizing the importance of having problem formulations and models to tackle real-world problems. Some variants that the authors present include heterogeneous fleet, multi-depot, time windows and asymmetric costs. Other real-world problem characteristics that have been addressed recently in the literature include representing users' inconvenience time [48] and modifying fixed routes with minimal disruption to customize pickup and drop-off locations [19].

The goal of this thesis is to define and solve two real-life inspired transportation problems: the Senior Transportation Problem (STP) and the Senior Transportation with Overbooking (STPOB). There are three levels of decisions in the STP:

1. the selection of requests to fulfil,
2. the assignment of requests to vehicles, and
3. the routing of vehicles.

In the STPOB, since all requests must be satisfied, the first decision is omitted. Each of these levels is a well-studied hard problem on its own and instances of reasonable size usually require sophisticated models or heuristic approaches to be solved.

In this chapter, we present an overview of problems relevant to the STP and the STPOB together with their associated solution techniques. More specifically, we focus on the literature surrounding the PDP and its variants, the Dial-a-Ride Problem (DARP), the Orienteering Problem (OP), the combination of the PDP and OP, and overbooking problems. We review both exact techniques and some heuristic algorithms that solve these related problems.

2.1 Related Problems

The vehicle routing problem with pickup and delivery or simply the PDP is a variation of the VRP in which sequence constraints are added to some subset of the visits. More specifically, a pickup location is required to be visited before a delivery location and both visits must be on the same route. Furthermore, the pickup and delivery problem with time windows (PDPTW) is the PDP with time window constraints associated with each location. In contrast, the OP is a variation of the VRP in which a fixed set of vehicles must visit as many locations as possible within the given time limit or given total route length.

The STP can be viewed as a combination of a PDPTW and an OP. In this section, we survey the recent work done on these problems. We present the variants of each problem that have been proposed and studied along with the techniques used to solve each variant, with a focus on exact techniques.

The STPOB is an extension of the STP which takes into account the overbooking aspect and minimizes the total expected cost when given the probability of cancellation of each request. We also provide review on the literature that presents problems with an overbooking condition.

2.1.1 Pickup and Delivery Problems with Time Windows

In 2008, Parragh et al. presented an extensive survey on PDP [52, 53]. Part I studies problems where the pickup or delivery is only at one single location (namely the depot) and part II surveys the class of problems with pickup and delivery requests between clients. Within this class, the problems are further categorized between paired pickup and deliveries where every pickup is paired with exactly one delivery and unpaired requests where multiple commodities can be picked from up and delivered to different locations. The DARP is an example of the paired PDP which will be discussed later in this subsection. Another important feature of the PDP is the presence of time window constraints (PDPTW). A comprehensive survey was done by Cordeau et al. on the one-to-one PDPTW [16].

The PDPTW can also be seen as a variant of the VRPTW with the additional precedence constraints from the pickup and delivery. The PDPTW is a routing problem in which the goal is to minimize the total travel costs while satisfying all time window constrained pickup and delivery requests associated with each location. Additionally to the given set of available vehicles, a set of pickup and delivery requests which specify its size, pickup and delivery locations, and service time windows is provided. The objective is to find a set of feasible routes that satisfy each request exactly once while minimizing the cost of all routes. Dumas et al. discussed this problem in detail providing a mathematical formulation and a column generation approach [20]. Additionally, Lau et al. formulated a tabu search with a construction heuristic approach [44]. Many PDPTW methodologies and formulations are inherited from the VRPTW which has been surveyed by Cordeau et al. [17] and later by Kallehauge et al [37].

Formulations of the PDPTW

The PDPTW is most often modelled with a three-indexed formulation [15, 13, 57] that comes from the vehicle flow formulation of the VRPTW [37]. One of the advantages of this formulation is its compactness, allowing easy readability. However, the number of variables and constraints grows in a polynomial factor with the number of requests and this model may become very large for sizeable problems. The three indexed formulation of [57] for a PDPTW is given in (2.1) - (2.12) .

Let n denote the number of requests, then the set of service locations $\mathcal{N}$ is denoted as $[0, 1, ..., n, n+1, ...2n+1]$ where nodes 0 and $2n+1$ represent the origin and destination depots, while subsets $\mathcal{P} \in \mathcal{N}$ and $\mathcal{D} \in \mathcal{N}$ denote the pickup and delivery locations respectively. Every vehicle $k \in \mathcal{K}$ is associated with a capacity C and each location i is associated with a load q_i, service duration d_i, and a time window $[a_i, b_i]$ during which the service can be done. Between two locations i and j, the travel time is denoted by t_{ij} and the cost by c_{ij}. This formulation employs binary variables x_{ij}^k which are equal to 1 if vehicle k visits location i directly before location j or 0 otherwise. Variables B_i^k indicate the time when vehicle k begins service at location i and Q_i^k indicates the load of vehicle k after completing service at location i. Then, PDPTW can be modelled as follows:

$$\min \sum_{k\in\mathcal{K}} \sum_{i\in\mathcal{N}} \sum_{j\in\mathcal{N}} c_{ij} x_{ij}^k \tag{2.1}$$

$$\text{subject to} \sum_{k\in\mathcal{K}} \sum_{j\in\mathcal{N}} x_{ij}^k = 1 \qquad \forall i \in \mathcal{P} \tag{2.2}$$

$$\sum_{j\in\mathcal{N}} x_{ij}^k - \sum_{j\in\mathcal{N}} x_{n+i,j}^k = 0 \qquad \forall i \in \mathcal{P}, k \in \mathcal{K} \tag{2.3}$$

$$\sum_{j\in\mathcal{N}} x_{0j}^k = 1 \qquad \forall k \in \mathcal{K} \tag{2.4}$$

$$\sum_{j\in\mathcal{N}} x_{ji}^k - \sum_{j\in\mathcal{N}} x_{ij}^k = 0 \qquad \forall i \in \mathcal{P} \cup \mathcal{D}, k \in \mathcal{K} \tag{2.5}$$

$$\sum_{i\in\mathcal{N}} x_{i,2n+1}^k = 1 \qquad \forall k \in \mathcal{K} \tag{2.6}$$

$$B_j^k \geq (B_i^k + t_{ij}) x_{ij}^k \qquad \forall i \in \mathcal{N}, j \in \mathcal{N}, k \in \mathcal{K} \tag{2.7}$$

$$Q_j^k \geq (Q_i^k + q_j) x_{ij}^k \qquad \forall i \in \mathcal{N}, j \in \mathcal{N}, k \in \mathcal{K} \tag{2.8}$$

$$B_i^k + t_{i,n+i} \leq B_{n+i}^k \qquad \forall i \in \mathcal{P}, k \in \mathcal{K} \tag{2.9}$$

$$a_i \leq B_i^k \leq b_i \qquad \forall i \in \mathcal{N}, k \in \mathcal{K} \tag{2.10}$$

$$max\{0, q_i\} \leq Q_i^k \leq min\{C, C + q_i\} \qquad \forall i \in \mathcal{N}, k \in \mathcal{K} \tag{2.11}$$

$$x_i j^k \in \{0, 1\} \qquad \forall i \in \mathcal{N}, j \in \mathcal{N}, k \in \mathcal{K} \tag{2.12}$$

The objective function (2.1) minimizes the total travel cost. Constraint (2.2) ensures that each location is visited exactly once and Constraint (2.3) ensures that the pickup and delivery locations of a request are visited by the same vehicle. Constraints (2.4) to (2.6) ensure that each vehicle starts at the starting depot, visits locations and ends at the ending depot. The consistency of time window variables is enforced through Constraint (2.7) and capacity variables through Constraint (2.8). Pickup locations are made sure to be visited before delivery locations by Constraint (2.9). Inequalities (2.10) and (2.11) bound the time window and capacity variables.

In contrast, Lu and Dessouky proposed a compact 0-1 two-index formulation for the multiple vehicle pickup and delivery problem [47]. The authors build a single Hamiltonian tour from all routes and remove the k index from the x_{ij}^k variables from the previous model by introducing new binary variables b_{ij} that indicate whether node i is visited before j in the tour. New constraints are included to reason about vehicles by using b_{ij} variables. For example, the formulation uses the constraint $b_{i,2n+j} = b_{n+i,2n+j}$, where i is a pickup node, $n+i$ is its associated delivery node, and $2n+j$ is a vehicle depot node, to model that both pickup and delivery locations must be on the same vehicle. The constraint states that if a pickup location is visited before a depot node, then its associated delivery node must also be visited before that depot node. The authors also propose preprocessing to remove infeasible arcs. For example, going from a delivery node to its associated pickup node is prohibited. The authors present a branch-and-cut approach and four classes of valid inequalities, and showed that the proposed approach can solve instances of up to 5 vehicles and 25 customers to optimality.

Very recently, Furtado et al. argued that the two-index formulation proposed by Lu and Dessouky can only solve small instances, and introduced a new two-index formulation for the PDPTW [24] . The formulation is an extension of the classical two-index formulation of the VRPTW [18]. Instead of using the b_{ij} variables, the authors make use of v_i variables that indicate the index of the vehicle that visits location i. They showed good performance (solving 26 instances to optimality) compared to the classical three-indexed formulation (solving 15 instances) and against the Lu and Dessouky two-indexed model (solving 10 instances). The two-indexed formulation by Furtado et al. is given in (2.13) - (2.26) .

$$\min \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}} c_{ij}x_{ij} \tag{2.13}$$

$$\text{subject to } \sum_{i\in\mathcal{N}} x_{ij} = 1 \qquad \forall j \in \mathcal{P}\cup\mathcal{D} \tag{2.14}$$

$$\sum_{j\in\mathcal{N}} x_{ij} = 1 \qquad \forall i \in \mathcal{P}\cup\mathcal{D} \tag{2.15}$$

$$B_j \geq B_i + d_i + t_{ij} - M(1-x_{ij}) \qquad \forall i \in \mathcal{N}; j \in \mathcal{N} \tag{2.16}$$

$$Q_j \geq Q_i + q_j - M(1-x_{ij}) \qquad \forall i \in \mathcal{N}; j \in \mathcal{N} \tag{2.17}$$

$$a_i \leq B_i \leq b_i \qquad \forall i \in \mathcal{N} \tag{2.18}$$

$$max\{0, q_i\} \leq Q_i \leq min\{C, C+q_i\} \qquad \forall i \in \mathcal{N} \tag{2.19}$$

$$B_{n+i} \geq B_i + d_i + t_{i,i+n} \qquad \forall i \in \mathcal{P} \tag{2.20}$$

$$v_{n+i} = v_i \qquad \forall i \in \mathcal{P} \tag{2.21}$$

$$v_j \geq j \cdot x_{0j} \qquad \forall j \in \mathcal{P}\cup\mathcal{D} \tag{2.22}$$

$$v_j \leq j \cdot x_{0j} - n(x_{0j}-1) \qquad \forall j \in \mathcal{P}\cup\mathcal{D} \tag{2.23}$$

$$v_j \geq v_i + n(x_{ij}-1) \qquad \forall i,j \in \mathcal{P}\cup\mathcal{D} \tag{2.24}$$

$$v_j \leq v_i + n(1-x_{ij}) \qquad \forall i,j \in \mathcal{P}\cup\mathcal{D} \tag{2.25}$$

$$x_{ij} \in \{0,1\} \qquad \forall i \in \mathcal{N}; j \in \mathcal{N} \tag{2.26}$$

Similar to the three-index formulation, there are a set of homogeneous fleet of vehicles $\mathcal{K}$, each with a capacity of C and a set of pickup $\mathcal{P}$ and delivery $\mathcal{D}$ nodes, each with a demand (or load) q_i, service time d_i and time window $[a_i, b_i]$ during which the service must be completed. Furthermore, a travel cost

c_{ij} and a travel time t_{ij} are associated with each pair of locations i and j. This formulation uses binary variables x_{ij} which equal 1 if location i is directly visited before j and integer variables v_i which indicate the index of the vehicle that visits location i. Additionally, variables Q_i model the load of the vehicle after visiting location i and B_i model the time that a vehicle starts the service at location i.

The objective function (2.13) minimizes the total travel cost. Constraints (2.14) and (2.15) ensure that each location is visited exactly once. The time window and capacity variables are restricted via constraints (2.16) and (2.17) and bounded though constraints (2.18) and (2.19) respectively. Constraints (2.21) ensure the pickup location and its associated delivery location are visited by the same vehicle. The variables v_j and x_{ij} are linked by Constraints (2.22) to (2.25). Finally, Constraints (2.26) ensure that variables x_{ij} are bounded to 0 or 1.

Solution Techniques for the PDPTW

The PDPTW is an NP-hard problem [7, 45]. In order to achieve the scale of problem solving that is necessary, while many of the approaches seek heuristic solutions, the state-of-the art for exact approaches has been based on sophisticated mathematical programming methods such as branch-and-cut-and-price [57]. The work done by Ropke et al. [57] extends the branch-and-price algorithm proposed by Dumas et al [20]. The formulation is based on the set partitioning formulation of the VRPTW [18]. This formulation assumes a set of homogeneous vehicles $\mathcal{V}$ and a set of all feasible routes Ω satisfying all time window, capacity, and precedence constraints. Let X_r be binary variables that are 1 if route r is used and 0 otherwise. Given Ω and the set of locations $\mathcal{N}$ with $\mathcal{P} \in \mathcal{N}$ and $\mathcal{D} \in \mathcal{N}$ as defined previously, let a_{ir} be coefficients that are 1 if route r contains request $i \in \mathcal{N}$, and 0 otherwise. Finally, letting c_r denote the cost associated with each route r, the formulation is as follows.

$$\min \sum_{r \in \Omega} c_r X_r \tag{2.27}$$

$$\text{subject to} \sum_{r \in \Omega} a_{ir} X_r = 1 \qquad \forall i \in \mathcal{P} \tag{2.28}$$

$$\sum_{r \in \Omega} X_r = |\mathcal{V}| \tag{2.29}$$

$$X_r \text{ binary} \qquad \forall r \in \Omega \tag{2.30}$$

The objective function (2.27) minimizes the total cost of all routes. Each request can only belong to one route (2.28) and there are exactly $|\mathcal{V}|$ routes (2.29).

In the column generation algorithm, a restricted master problem considering a subset of Ω is solved. Additional columns, i.e., feasible routes, are generated by solving a pricing subproblem [57] that is given below.

$$\min \sum_{i,j \in \mathcal{N}} d_{ij} x_{ij} \tag{2.31}$$

$$\text{subject to } (2.3) - (2.12) \text{(dropping the index } k\text{)}$$

$$\text{where } d_{ij} \text{ is defined as: } d_{ij} = \left\{ \begin{array}{ll} c_i j - \pi_i & \forall i \in \mathcal{P}, j \in \mathcal{N} \\ c_{ij} & \forall i \in \mathcal{N} \setminus \mathcal{P}, j \in \mathcal{N} \end{array} \right\} \tag{2.32}$$

and π_i is the dual variable associated with the set partitioning Constraints (2.28).

The set partitioning approach was shown to work for instances of sizes up to 500 requests with tight time windows [20]. Instances with less constrained time windows however, are still very hard to solve. This approach was re-implemented by Ropke et al. [57] with more modern technology and reportedly solved less constrained instances of sizes up to 100 requests under 1200 seconds.

More recently, Baldacci et al. presented a new exact algorithm for the PDPTW which is also based on the set-partitioning formulation [3]. The authors describe various procedures including two ascent heuristics to find near-optimal dual solutions of the LP-relaxation. The dual solution is then used to generate a reduced problem in which only routes that have reduced costs than a specific threshold are considered. The reduced problem is then solved using an integer programming solver. These approaches were previously proposed by the same authors to solve the capacitated VRP problem [4]. The authors have claimed that this approach is comparable to the algorithm proposed by Ropke et al. [57] and even solves 15 previously open instance.

There are two main streams of heuristics that exist for the VRP and PDPTW [42, 14]. One gradually assigns locations to vehicle routes, whereas the other assigns each location to a different route, then merges routes based on some *savings criterion*. Within the constructive heuristics, many algorithms reason about the geographical location of the locations to visit or the euclidean distance of the locations from the central depot. These include the *sweeping algorithm* [29] and the *petals algorithm* [5].

Dial-a-Ride Problems

In the PDPTW, vehicles transport goods, however in the STP and STPOB, vehicles transport people, resulting in a set of user inconvenience constraints such as maximum ride time. The version of PDPTW in which people are transported is known as the Dial-a-Ride Problem (DARP) [15] or the Handicapped Persons Transportation Problem [62]. This section briefly outlines the different exact solution techniques that are available to solve the DARP.

The DARP has been widely studied and its variants and solution techniques are explored in various survey papers [53, 15]. Cordeau et al. [15] note that there could be another objective of maximizing satisfied demand or quality of service but do not provide any formulation or reference to any work that has been done in this area, and to the best of our knowledge, no literature has looked at a DARP with this second objective.

Many researchers have also looked at real-life applications of the DARP and real-world inspired constraints. For example, in 2009, Gørtz et al. presented approximation methods for the preemptive multi-vehicle DARP in which users are allowed to be transferred at intermediate locations and served by multiple vehicles [30]. Furthermore, in 2011, Kim and Haghani [38] modelled the time-varying travel time and multi-depot DARP in a MIP formulation and provided three different heuristics to solve this

problem.

Other Variants of the PDPTW

Sigurd et al. proposed the pickup and delivery problem with time windows and precedence constraints (PDPTWP) that is inspired by the transportation of live animals where specific farms must be visited after other locations [60]. The authors provided a Dantzig-Wolfe decomposition approach that splits the problem into a master problem that is similar to a set covering problem and a subproblem that is a routing problem. The LP relaxation of the decomposed problem is solved through column generation. The authors presented real-life instances of transporting live pigs and showed that instances up to 580 nodes could be solved to optimality.

Another variant of the PDPTW is the multi-commodity one-to-one pickup and delivery traveling salesmen problem. In this problem which does not consider time window constraints, the vehicle must visit all locations exactly once. Some of the locations are associated with pickup and delivery activities and the vehicle has a fixed capacity. The current state-of-the-art algorithm is a Benders Decomposition algorithm presented by Hernandez-Perez et al. [33].

Additionally, Irnich has looked at the PDPTW with multiple depots and a single hub from a set covering and a knapsack view [35]. In this problem, the routes are often short with large demand size and tight time windows, thus enumerating all vehicle/route combinations is possible. Decomposition models were presented and were shown to solve medium size instances to optimality. In contrast, Contardo et al. employ a branch-and-price framework to solve a more general multi-depot VRP (MDVRP) [12] where there is no restriction on routes or time window characteristics. Valid inequalities were added to strengthen the formulation and the authors showed through extensive computational experiments that their algorithm is competitive with the state-of-the-art algorithms. However, the pickup and delivery constraints were not captured in the MDVRP.

2.1.2 Orienteering Problems

The Orienteering Problem (OP), also known as the selective travelling salesman problem [43] or the traveling salesman problem with profits [22], is another variant of the routing problems. In this problem, a person must visit locations, however, each location is associated with a "prize" or "profit" and the goal is to find a path through a subset of the locations in order to maximize the total profit collected from all locations while satisfying some tour length or travel time limit constraints. Given a set of vertices $\mathcal{N}$, where vertex 1 is the starting vertex and vertex N is the ending vertex, each with a profit or score S_i, travel time between vertex i and j, t_{ij}, and a time limit T_{max}, the OP can be modelled as an integer program. Binary variables x_{ij} that are 1 if vertex i is visited directly before vertex j in the tour or 0 otherwise and variables u_i indicating the position of vertex i in the tour are used.

$$\max \sum_{i=1}^{N-1} \sum_{i=2}^{N} S_i x_{ij} \tag{2.33}$$

$$\text{subject to } \sum_{i=2}^{N} x_{1j} = \sum_{i=1}^{N-1} x_{iN} = 1 \tag{2.34}$$

$$\sum_{i=1}^{N-1} x_{ik} = \sum_{i=2}^{N} x_{kj} \leq 1 \qquad \forall k = 2, ..., N-1 \tag{2.35}$$

$$\sum_{i=1}^{N-1} \sum_{i=2}^{N} t_{ij} x_{ij} \leq T_{max} \tag{2.36}$$

$$2 \leq u_i \leq N \qquad \forall i = 2, ..., N \tag{2.37}$$

$$u_i - u_j + 1 \leq (N-1)(1 - x_{ij}) \qquad \forall i, j = 2, ..., N \tag{2.38}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i, j = 1, ..., N \tag{2.39}$$

$$u_i \in \mathbb{Z} \qquad \forall i = 2, ..., N \tag{2.40}$$

The objective function (2.33) maximizes the total profit collected. Constraints (2.34) ensure that the tour starts at the starting vertex and ends at the ending vertex. Constraint (2.35) guarantees that each vertex is visited at most once and that there is constant flow through all vertices. The maximum tour length is restricted by Constraint (2.36). Constraints (2.37) and (2.38) bound the u_i variables and link them to the tour positions respectively. Finally, Constraints (2.39) make sure that all x_{ij} variables are binary.

The OP is extensively defined and reviewed in survey papers [22]. The current state-of-the-art exact technique to solve the OP is a branch-and-cut algorithm proposed by Fischetti et al. [23]. All current research is focused on heuristic-based algorithms including tabu search [27], genetic algorithm [61], and ant colony optimization approach [46].

The variation of the OP that incorporates multiple vehicles is named the team orienteering problem (TOP) in survey papers by Vansteenwegen et al. and Gunawan et al. [66, 31]. Vansteenwegen et al. [66] included descriptions of problem instances, benchmark instances, and solution approaches; whereas, Gunawan et al. [31] provide a more recent and comprehensive survey on many variants of the OP and TOP: TOP with time windows, generalized OP, etc. Exact solution techniques to solve the TOP include column generation [9] and branch-and-cut-and-price [26] which is similar to the algorithm for the PDPTW [57]. Recently, Gedik et al. developed a constraint programming approach to solve the TOP and have found one new best-known solution [25].

Other variations of the OP and TOP include the capacitated (T)OP and (T)OP with time windows. The capacitated (T)OP is an OP where the vehicles have a fixed capacity for the sizes of the items they collect from the locations. Archetti et al. proposed exact and heuristic methods for solving the capacitated TOP and capacitated profitable tour problems and showed that the heuristics demonstrate very good solutions and often find optimal solutions [1]. In contrast, the (T)OP with time windows imposes time window constraints on all locations. Boussier et al. presented a branch-and-price framework for the TOP with time windows [8]. The problem instances with 10 vehicles and 100 locations were solved to optimality in under 800 seconds. Since this problem is very hard, most studies proposed heuristic-based methods. The current best performing algorithms are ant colony optimization [49] and iterated

local search [65]. These approaches could not be compared directly as the problem instances tested were different, however, the ant colony optimization has shown to provide good quality results, whereas the iterated local search is able to provide solutions quickly.

2.1.3 Combination of the PDP and OP

Though the PDP and the OP have been studied extensively, a hybrid of the two problems has only been looked at by two groups. Baklagis et al. [2] have proposed a branch-and-price framework to tackle the team orienteering pickup and delivery problem with times windows and Qiu et al. [54] have proposed a graph search and a maximum set packing formulation that is specially tailored for homogeneous fleets to solve the profit-maximizing multi-vehicle pickup and delivery selection problem.

Qiu et al. [54] presented a pure MIP model while Baklagis et al. [2] only presented their branch-and-price framework. Qiu et al.'s MIP model is based on Ropke et al.'s MIP model [57] that is the standard three-index formulation for solving the PDPTW presented in Section 2.1.1. Qiu et al.'s model is presented in (2.41) - (2.52).

This problem is given a set of vehicles $\mathcal{K}$, each with a capacity Q^k and a set of pickup locations $\mathcal{P}$ and delivery locations $\mathcal{D}$, each with a service time d_i, profit π_i, load size q_i, and a time window $[a_i, b_i]$. The travel time between two locations is t_{ij} and the travel cost is c_{ij}. This formulation uses binary variables x_{ij}^k that are equal to 1 if vehicle k travels to location i directly before location j, variables v_i^k that indicate when vehicle k leaves location i, and variables w_i^k that indicate the load of vehicle k when leaving location i.

$$\max \sum_{k\in\mathcal{K}} \left(\sum_{i\in\mathcal{P}} \pi_i \sum_{j\in\mathcal{V}} x_{ij}^k \right) - \sum_{k\in\mathcal{K}} \sum_{i\in\mathcal{V}} \sum_{j\in\mathcal{V}} c_{ij} x_{ij}^k \tag{2.41}$$

$$\text{subject to } \sum_{i\in\mathcal{V}} x_{0,j}^k = \sum_{i\in\mathcal{V}} x_{i,2n+1}^k = 1 \qquad \forall k \in \mathcal{K} \tag{2.42}$$

$$\sum_{i\in\mathcal{V}} x_{i,2n+1}^k \leq 1 \qquad \forall k \in \mathcal{K} \tag{2.43}$$

$$\sum_{j\in\mathcal{V}} x_{ij}^k - \sum_{j\in\mathcal{V}} x_{n+i,j}^k = 0 \qquad \forall i \in \mathcal{P}, k \in \mathcal{K} \tag{2.44}$$

$$\sum_{j\in\mathcal{V}} x_{ji}^k - \sum_{j\in\mathcal{N}} x_{ij}^k = 0 \qquad \forall i \in \mathcal{P} \cup \mathcal{D}, k \in \mathcal{K} \tag{2.45}$$

$$v_j^k \geq (v_i^k + t_{ij} + d_i) x_{ij}^k \qquad \forall i \in \mathcal{V}, j \in \mathcal{V}, k \in \mathcal{K} \tag{2.46}$$

$$v_i^k + t_{i,n+i} \leq v_{n+i}^k \qquad \forall i \in \mathcal{V}, j \in \mathcal{V}, k \in \mathcal{K} \tag{2.47}$$

$$a_i \leq v_i^k \leq b_i \qquad \forall i \in \mathcal{V}, k \in \mathcal{K} \tag{2.48}$$

$$v_{2n+1}^k - v_0^k \leq T \qquad \forall k \in \mathcal{K} \tag{2.49}$$

$$w_j^k \geq (w_i^k + q_j) x_{ij}^k \qquad \forall i \in \mathcal{V}, j \in \mathcal{V}, k \in \mathcal{K} \tag{2.50}$$

$$\max\{0, q_i\} \leq Q_i^k \leq \min\{Q^k, Q^k + q_i\} \qquad \forall i \in \mathcal{V}, k \in \mathcal{K} \tag{2.51}$$

$$x_{ij}^k \in \{0, 1\} \tag{2.52}$$

The objective function (2.41) maximizes the total profit collected from all visits. Similar to the three-indexed model (see Section 2.1.1), Constraints (2.42) to (2.45) are constraints regarding the connectivity

of routes and precedence of nodes. Constraints (2.46) to (2.49) are the time window constraints, while (2.50) and (2.51) are the load and capacity constraints respectively. Constraints (2.52) bound the variable x_{ij}^k to binary.

A summary of the different problem variants and their respective state-of-the-art exact techniques are shown in Table 2.1. It is important to note that the size of a problem instance solved can be sometimes misleading as a measure of the effectiveness of an algorithm because the problem might be very constrained and data dependent. Thus, there can be multiple current state-of-the-art algorithms depending on the measure. Furthermore, the list is relatively short since most research has focused on heuristic methods as these problems are often too hard and the efficiency of exact methods is very poor. It can also be noted from the list that no other VRP variant has considered all the problem characteristics present in the STP.

2.1.4 Problems with Overbooking

In the STPOB, we allow requests to be overbooked and in this section, we review some work on overbooking strategies in various industries including airlines [11], hotels [40] and healthcare [50, 59].

The notion of overbooking or overselling was introduced to the airline industry as early as early as 1958 [6] to compensate for the waste of resources from cancelled reservations [21]. Most of the work is focused on stochastic models and policy making. Recently, some researchers in operations research have studied this problem as an optimization problem in a clinical setting. Samorani et al. provided a stochastic program that finds optimal schedules given an estimated probability of appointment show ups to improve overbooking in clinic appointment scheduling [59]. This model employs the slot compression technique [39], which allows time overlap in all appointments. Another application of overbooking is in hotel revenue management where the length of stay is variable and changes from customer to customer. Lai et al. proposed a robust network optimization model in an uncertain environment where both demand and stay length are stochastic [40]. In the healthcare appointment overbooking problem, each patient can only be appointed to a subset of doctors similar to in the STPOB, where a particular request can only be assigned to a subset of vehicles due to capacity constraint. The hotel overbooking problem considers a stochastic length but does not consider limitations of assignment. However, both problems fail to consider strict time window constraints and travel time and cost between the different appointments/stays/requests that the STPOB requires.

To the best of our knowledge, the notion of overbooking has not been applied to transportation problems such as the VRP which assumes that all locations are visited and no cancellation occurs.

2.2 Conclusion

Through a thorough study of the VRP literature, we have noticed that, recently, there has been a tendency to focus on real-world inspired problem definitions and constraints. Many researchers have partnered with organizations to understand the true problem difficulties. Similarly, we have partnered with a non-profit organization Community and Health Assistance to Seniors (CHATS) to create new problem definitions of the STP and STPOB. We position our research work as a novel problem definition that comprises problem characteristics from both the PDPTW such as time windows, pickup and deliveries, and routes and from the OP such as the selectivity and maximization of scores. In this chapter, we have presented recent works on problems including the PDPTW, DARP, and OP along with their

Table 2.1: Summary of STP Related Problems

Authors	Year	Algorithm	Variant	Problem Size		Runtime	Constraints							
				Veh.	Req.	(sec)	TW	W	PD	MD	Sel	MRT	Het.	Cap.
This work	2017	MIP	STP	5	20	<600	✓	✓	✓	✓	✓	✓	✓	✓
		CP	STP	270	385	<600								
		MIP/CP LBBD	STP	238	226	<600								
		CP/CP LBBD	STP	20	50	<600								
Qiu et al. [54]	2016	Graph Search	PPDSP	1	300	<550	✓	✓	✓		✓			✓
		Maximum Set Packing	PPDSP	30	500	<4300								
Baklagis et al. [2]	2016	B&P	TOPDPTW	5	20	<200	✓	✓	✓		✓			
Dumas et al. [20]	1991	CG	PDPTW	NA	55	<313	✓		✓	✓			✓	✓
Sigurd et al. [60]	2004	Delayed CG, B&B	PDPTWP	NA	205	<1807	✓		✓	✓			✓	✓
Ropke et al. [57]	2009	B&C&P	PDPTW	NA	500	<5000	✓		✓					✓
Baldacci et al. [3]	2011	CG	PDPTW	solved 15 open instances			✓		✓	✓			✓	✓
Cordeau et al. [13]	2006	B&C	DARP	NA	24	<240	✓		✓			✓		✓
Hernandez-Perez et al. [33]	2009	BD	mPDTSP	1	25	<1800			✓					✓
Archetti et al. [1]	2007	B&P	CTOP, CPTP	15	151	<1866					✓			✓
Butt and Ryan	1999	CG	TOP	15	100	<16708		✓			✓			
Boussier et al. [8]	2007	B&P	TOP,	4	102	<1320		✓			✓			
			SVRPTW	10	100	<798	✓	✓			✓			
Contardo et al. [12]	2014	Cutting planes and CG	MDVRP	solved 14 open instances						✓				

Veh.: Number of vehicles, Req.: Number of requests
TW: time windows, W: weighted requests, PD: pickup and delivery, MD: multi-depot, Sel.: selective requests, MRT: max ride time, Het.: heterogeneous fleets, Cap.: limited vehicle capacities
STP: Senior transportation problem, PPDSP: profit-maximizing pickup and delivery selective problem, TOPDPTW: team orienteering pickup and delivery problem with time windows, PDPTW: pickup and delivery problem with time windows, DARP: dial-a-ride problem, mPDTSP: multi-commodity pickup and delivery travelling salesman problem, MDVRP: multi-depot vehicle routing problem, CTOP: capacitated team orienteering problem, CPTP: capacitated profitable tour problem, TOP: team orienteering problem, SVRPTW: selective vehicle routing problem with time windows, PDPTWP: Pickup and Delivery Problem with Time Windows with Precedence Constraints
MIP: Mixed Integer Programming, CP: Constraint Programming, LBBD: Logic-based Benders Decomposition, CG: Column Generation, B&B: Branch and Bound, B&P: Branch and Price, B&C&P: Branch and Cut and Price, BD: Benders decomposition

different formulations and solution techniques. The combination of the PDPTW and OP has been only looked at by two groups. However, these problems do not address the multi-depot, heterogeneous fleets, and maximum ride time aspects that our problems include. Therefore, we believe that the definitions of the STP and STPOB will bring novelty and inspire future research.

Most of the state-of-the-art techniques for these problems involve sophisticated mathematical programming methods including branch-and-cut-price algorithms. However, the use of constraint programming or the hybrid of MIP and CP in a logic-based Benders decomposition framework have not been explored.

Chapter 3

Data Generation and Analysis

In order to test the different solution techniques that are presented in this thesis, we have generated 75 random instances as well as extracted 280 problem instances from real-world data provided by a partnering organization. This chapter explains the process of data generation and analyses of the real world data with the goal of retrieving insightful information as well as a better understanding of the true nature and the difficulty of the problems studied.

3.1 Generated Problem Instances

In the generated problem instances, we varied the size of the problem, i.e., the number of requests and vehicles and the sizes of the time window (TW) of each request and vehicle. Table 3.1 outlines the five different problem sizes with varying number of vehicles and requests.

Table 3.1: Generated dataset sizes.

# of Vehicles	# of Requests
2	6
3	10
5	20
10	30
20	50

In addition to varying problem sizes, we also experimented with three different time window sizes: big, normal and small. The time unit has been set to 1 and the maximum time horizon has been set to 900. Furthermore, the maximum ride time for all requests has been set to be 120 time units. These values are chosen based on real life situations. Time windows for all locations (starting and ending depots of vehicles, and pickup and delivery locations of requests) are set between 0 to 900 with varying window length. Big time windows allow 600 to 900 time units, normal allows 180 to 360 time units and small allows 80 to 180 time units. A time window length, l_i, is randomly generated for each location i. Additionally, each location i is paired with another location i': starting depots are paired with ending depots, and pickup locations are paired with delivery locations. Let d_i denote the time it takes to travel between paired locations i and i'. For a starting depot or a pickup location i, the earliest start time est_i is then randomly generated between 0 and $\max(0, 900 - l_i - d_i)$ and the latest finish time lft_i is

simply $(est_i + l_i)$. For an ending depot or a delivery location i', the earliest start time $est_{i'}$ is randomly generated between $(0 + d_{i'})$ and $max(0 + d_{i'}, 900 - l_{i'})$ and the latest finish time $lft_{i'}$ is $(est_{i'} + l_{i'})$. Accordingly, the time window for a location i is $[est_i, lft_i]$. For each combination of problem size and time window type, five datasets have been generated, making in total 75 instances.

Each dataset contains both vehicle information and request information. For each vehicle, a capacity between 2 and 6, and start depot and end depot service times between 2 and 16 time units have been created following a uniform distribution. These features also mimic real life situations where a vehicle can hold 2 to 6 passengers and can spend 2 to 16 time units at depot locations. Each request has a size between 1 and 3, a weight between 1 and 5, and pickup location and delivery location service times between 2 and 16. Finally, a transition matrix between all locations (starting depots, ending depots, pickup locations, delivery locations) is generated. Each location has associated random coordinates which cover a squared area of 120 km by 120 km. Euclidean distances between each pair of locations are calculated from the coordinates and transformed to travel time with a speed factor of 60 km/h to form the transition matrix. Euclidean distances are used in order to ensure the triangular inequality between locations which means that given locations A, B, and C, $TravelTime(A, C) \leq TravelTime(A, B) + TravelTime(B, C)$. The lower bound and upper bound for the characteristics of the datasets are summarized in Table 3.2. All numbers are generated randomly using a uniform distribution.

Table 3.2: Generated datasets' bounds on problem characteristics.

Characteristic	Lower Bound	Upper Bound
Vehicle		
capacity	2	6
start deport and end depot service time	2	16
Request		
size	1	3
weight	1	5
pickup location and delivery location service time	2	16
travel time	1	60

The Senior Transportation Problem with Overbooking (STPOB) requires additional information including the cost and the cancellation probability of each trip. The cancellation probability is randomly generated using a skewed uniform distribution with a mean of 0.2 as the partnering organization roughly predicts the cancellation percentage to be at 20%. The cost of each trip depends on both the direct distance of the trip and the type of driver, paid driver or volunteer driver. A quarter of all vehicles are assumed to be paid drivers and they are given a cost factor of 2 while other vehicles are given a cost factor of 1.5.

3.2 CHATS Problem Instances

Community & Home Assistance to Seniors (CHATS) is a non-profit organization that provides senior transportation services in the north of Toronto area. From the CHATS database, we extracted 72,883 entries of requests and 52,494 records of vehicles over a total of 280 operating days starting from January 1, 2015 and ending in January 27, 2016. From these records, a total of 280 instances were created. This

section first describes the extracted data, then explains the prediction model used to generate cancellation probabilities for the STPOB.

3.2.1 Description of Data

Three major categories of information have been pulled from the database:

1. clients' demographic information,
2. request related information, and
3. vehicle related information.

These elements were selected based on the need for data analysis as well as for use in building the prediction model. For this specific time period, 67% of the clients were female, 31% were male and 2% of the gender information was missing. On average, female clients were aged at 79 years old and male clients were aged at 75 years old. The summary of the elements along with their mean, standard deviation, and some other basic statistical components are presented in Table 3.3. Additionally, the distribution of all requests' earliest pickup time, latest pickup time, earliest delivery time and latest delivery time is shown in Figure 3.1.

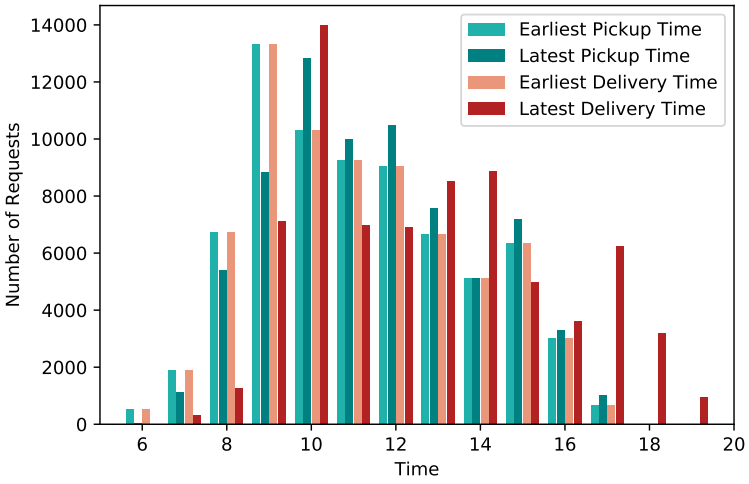


Figure 3.1: Distribution of requests' earliest and latest pickup and delivery times.

Requests Information

On average, there are 260 requests per day and the maximum number of requests per day is 554 which occurred on July 29, 2015. Each request is labelled with three possible statuses: scheduled, unscheduled, or cancelled. Unscheduled requests are often due to unavailable vehicles or drivers, or due to the organization's scheduling, while cancelled requests are scheduled requests that are subsequently cancelled by the client. The average number of scheduled, unscheduled and cancelled requests are respectively 196, 18, and 47 per day. The total number of requests in the three categories are graphed in Figure 3.2.

We can observe a steep peak around the end of July 2015 but when examining the data, there does not seem to exist any unusual entry, thus it could simply be an outlier. There is also a large drop by the

Table 3.3: Elements of the CHATS dataset.

Variable	mean	std	min	25%	50%	75%	max	Count	Pct
Client Information									
Gender									
- Female								48596	67%
- Male								22656	31%
- NA								1631	2%
Age	78.02	12.65	21	71	80	86	115		
- Female	79.08	10.93	44	73	81	86	115		
- Male	75.26	13.22	24	68	77	84	115		
- NA	100.24	15.63	55	88	102	115	115		
Preferred Language									
- English								38802	53%
- Other								33283	46%
- NA								798	1%
Request Information (72883)									
Size (number of seats needed)	1.21	0.41	1	1	1	1	2		
Direct Travel Distance (km)	11.86	11.57	0.00	3.94	8.70	14.93	102.91		
Fare to Collect ($)	6.81	9.48	0	0	0	12	100		
Weight	7.24	3.82	1	6	6	8	15		
Pickup Service Time (min)	0.23	0.50	0	0	0	0	5		
Delivery Service Time (min)	0.23	0.50	0	0	0	0	5		
Pickup TW Duration (hr)	0.42	0.17	0.00	0.33	0.33	0.67	15.33		
Delivery TW Duration (hr)	1.41	0.80	0.00	0.67	1.08	2.33	8.08		
Vehicle Information (52494)									
Capacity	3.31	3.59	1	1	2	4	17		
Depot Start Time	09:11	2.19	00:00	08:00	08:30	09:00	19:30		
Depot End Time	16:42	1.95	00:00	16:30	16:30	18:00	22:00		
Duration (hr)	7.52	2.56	0	6.5	8	8.75	16		

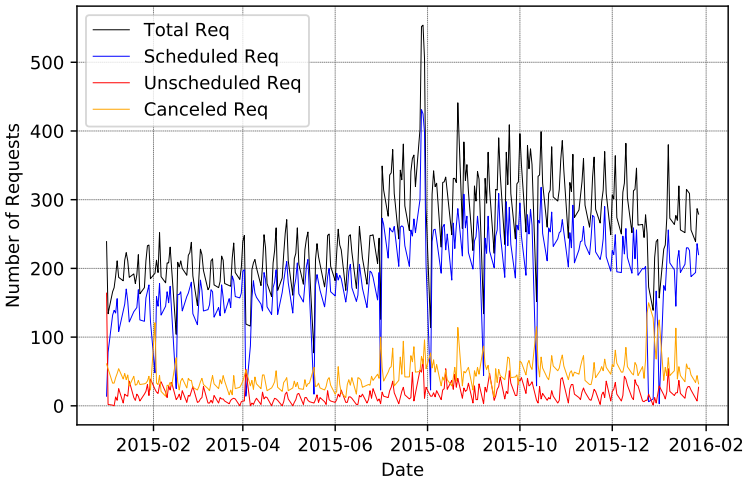


Figure 3.2: Number of scheduled, unscheduled, cancelled and total requests per day.

end of December 2015, around Christmas and New Year's time. It is natural to expect a big decrease in the number of requests during holidays. Furthermore, there also seem to be periodic drops and when examining the dates closely, these are all national or provincial holidays, thus again, the decrease is expected. Figure 3.3 presents the proportion of scheduled, unscheduled and cancelled requests per day and it can be seen that the percentage of the three categories is relatively constant. During the summer, when there are higher demands, the percentage of unscheduled requests seems to increase a little. And as expected, whenever there is a holiday, the percentage of cancelled requests increases significantly.

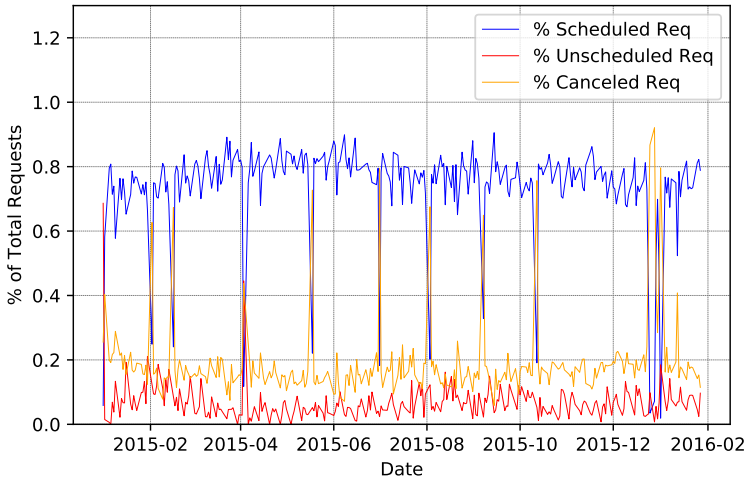


Figure 3.3: Percentage of scheduled, unscheduled, cancelled and total requests per day.

Overall, there seems to exist a big increase in demand during summer that slowly reduces over the winter. However, since the following year's data is not available, we cannot predict whether the demand will continue to drop or maintain the same level. Figure 3.4 shows the aggregated number of requests for each month where the increase can be clearly seen.

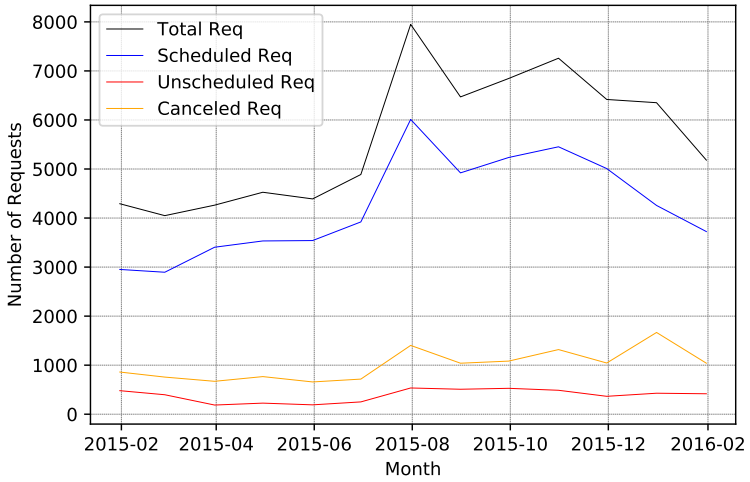


Figure 3.4: Number of scheduled, unscheduled, cancelled and total requests per month.

When zooming onto the data for a shorter period of time, we can still observe periodic small rises and drops. There seems to be an increase in demand at the end of the week and demand is much lower at the beginning of the week as shown in Figure 3.5.

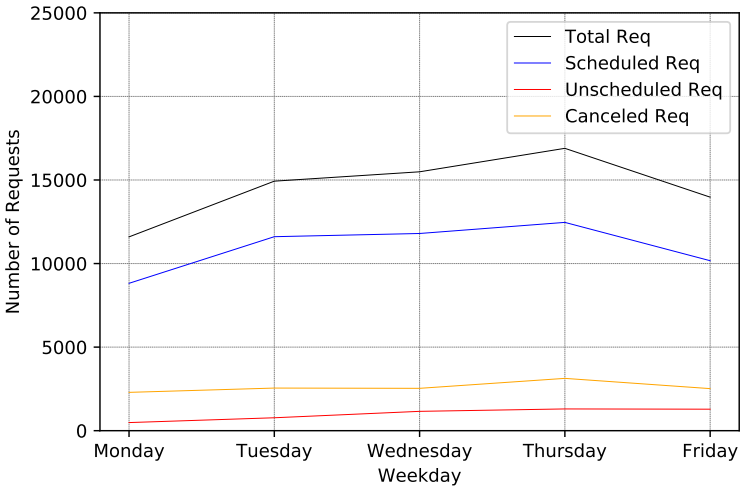


Figure 3.5: Number of scheduled, unscheduled, cancelled and total requests per weekday.

Vehicle Information

CHATS has two types of drivers: paid drivers who drive CHATS vehicles and get paid $18 per hour and volunteer drivers who use their own vehicles and get paid $0.05 for each kilometre. Assuming an average speed of 60 km/h, a paid driver gets paid $0.3 per km. In addition to the pay for paid drivers, CHATS reimburses the fuel cost which averages $0.14 per km for paid drivers only. Overall, the cost factor of a paid drive is $0.44/km while the cost factor for a volunteer driver is $0.05/km. The taxi cost factor is then computed to be $1.75/km. There are on average 187 vehicles available each day where 87% are volunteer drivers and 13% paid drivers. The daily trend of available vehicles is shown in Figure 3.6.

The trend of vehicles is similar to that of the requests with the outliers in end of July 2015 and the large increase over the summer time. However, there does not seem to be any significant decrease in vehicle availabilities during holidays, and the weekly rise and drop are still observed. The percentage of paid and volunteer drivers per day is shown in Figure 3.7. We can observe that the organization started with a 80-20 split but had a significant increase in the number of volunteer drivers around July 2015 while the number of paid drivers has been relatively constant.

Depending on the model of the car, each vehicle has a different capacity. From the CHATS dataset, the smallest capacity is 1 which means that the vehicle can only serve one client, while the largest capacity vehicle is a small bus that can take up to 17 people. The distribution of vehicle capacity for both paid and volunteer drivers are shown in Figure 3.8.

3.2.2 Cancellation Probability Prediction Model

In the current database, we only have a flag that indicates whether a request was cancelled or not. Yet, we wish to predict a probability of cancellation based on some features of a given request. Therefore, we

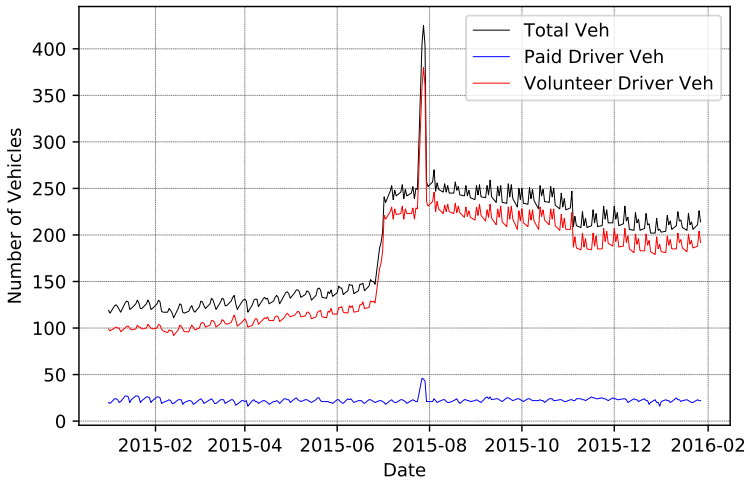


Figure 3.6: Number of paid and volunteer drivers per day.

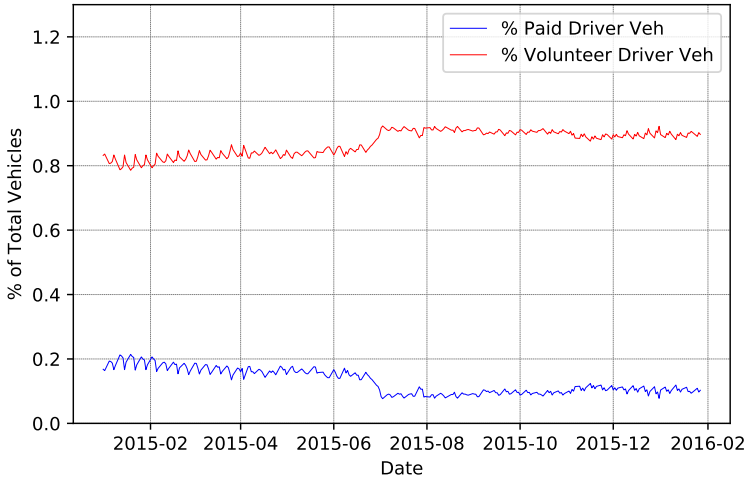


Figure 3.7: Percentage of paid and volunteer drivers per day.

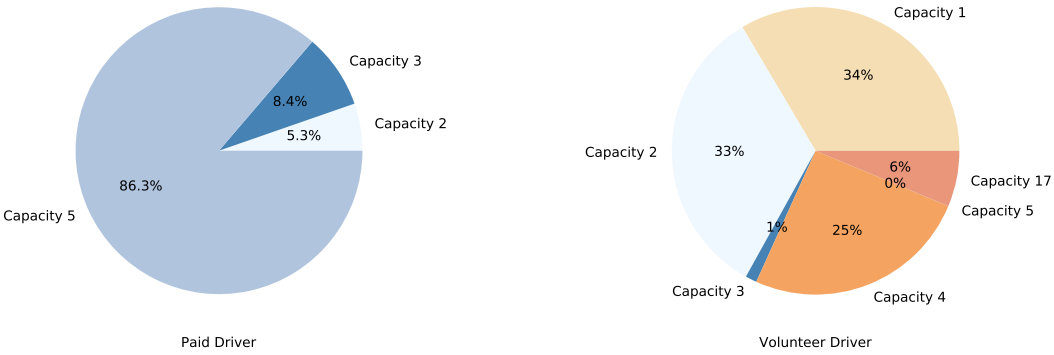


Figure 3.8: Distribution of vehicle capacity for paid and volunteer drivers.

built a model to predict the probability of cancellation. We have identified 9 features that are readily available and that will be used towards the prediction model. These features are: age, gender, language preference of the client, weekday, month, weight, distance, fare, and size of the request. There are other features such as weather that could also affect the probability of cancellation, however, due to the time limit of the project and the focus of this thesis being optimization methods, they have not been included.

We have looked into the categorical features including age and gender of the client and the weight of each request to compute the average cancellation probability in each category. Figure 3.9 shows the number of clients in each age category and the probability of cancellation of each age category. Similarly, the information for gender of the client is shown in Figure 3.10. Figure 3.11 presents the information on the different weight categories. The bars show how many requests fall into each weight category and the orange ticks show the weight associated with each weight category.

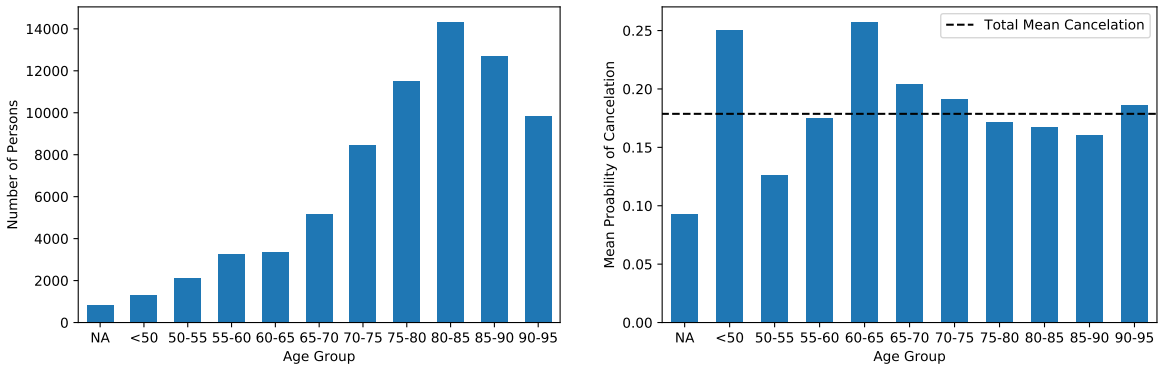


Figure 3.9: Cancellation by age of client.

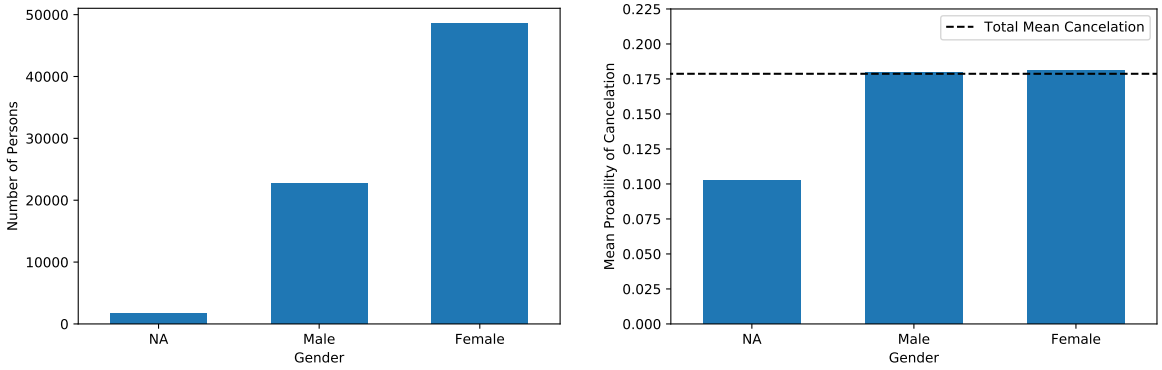


Figure 3.10: Cancellation by gender of client.

The training set comprises of 75% of the request records, totalling 54531 records, and the test set holds 18352 records. The target variable is the cancellation flag with 0 being cancelled and 1 being not cancelled. A few elementary prediction models have been then fit to the datasets. The models' performance summary is shown in Table 3.4. Python 3.6 has been used for data cleaning, feature extraction, and modelling. The python machine learning package **SCIkit-Learn v9.0** (http://scikit-

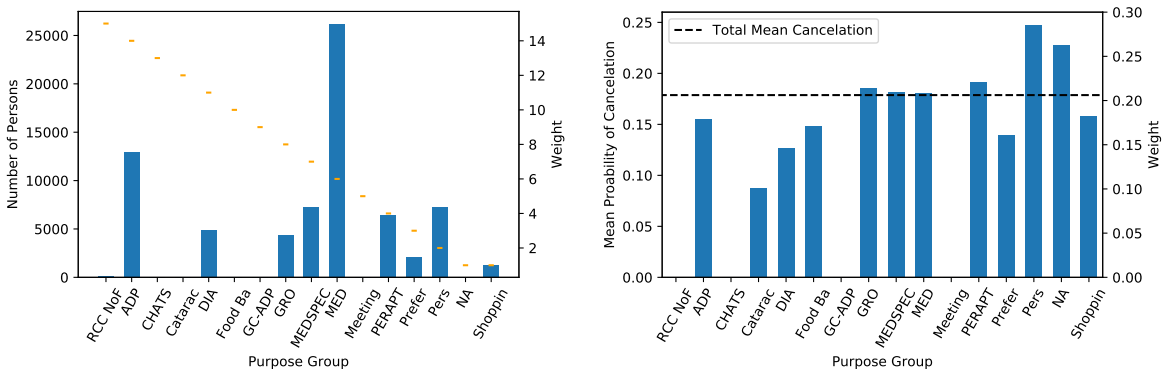


Figure 3.11: Cancellation by weight of request.

learn.org/) was used for the Random Forest, Extra Tree Classifier, Decision Tree Classifier, Ada Boost, Gradient Boosting Classifier, Boosting Regressor and Logistic Regression models.

Table 3.4: Prediction models performance summary.

	AUROC	MSE
Ensemble		
Random Forest	0.7073	
Decision Tree	0.6710	
Extra Tree	0.6822	
Ada Boost	0.5876	
Gradient Boosting Classifier	0.5872	
Regression		
Gradient Boosting Regressor		0.1453
Logistic Regression	0.5505	

For the four ensemble models and the Logistic Regression model, the area under the receiver operating characteristic curve (AUROC) was measured. The AUROC measures the proportion of true positive rate against false positive rate and is equal to the expected true positive rate. For example, an AUROC value of 0.8 means that in 80% of the time, the model will return a true positive prediction. A value of 1 is a perfect prediction model and a value of 0.5 is a worthless prediction model. The Mean Squared Error (MSE) for the Gradient Boosting Regressor measures the squared errors between the actual data point and the fitted line where 0 means a perfect fit. Based on these results, it can be seen that none of the models accurately predicted the cancellation probability. However, since the goal of this exercise is to compute a probability of cancellation, we have chosen the best performing model, the random forest method, to compute the probability of cancellation that is associated with each request of the CHATS dataset. This probability is then added to the CHATS problem instances as input to test our algorithms for the STPOB.

3.3 Summary

We have randomly generated 75 instances with varying time window sizes and varying number of requests and vehicles following a uniform distribution. The number of requests varies from 6 to 50 while the number of available vehicles varies from 2 to 20 and time window lengths varies from 80 time units to 900 time units out of a time horizon of 900 units. We have also extracted 280 instances from the CHATS database and performed basic data analysis. Seasonality and periodic shift in the number of requests and available vehicles have been observed. On average, there are 260 requests and 187 available vehicles per day. The problem sizes for the CHATS instances are much larger in both the number of requests and the number of vehicles compared to the generated instances. Additionally, for the CHATS instances, the duration of time windows for all requests is around 1.4 hours and does not vary vastly, while the duration of the availability of vehicles varies significantly and has an average of 7.5 hours. We have also constructed multiple prediction models to predict the cancellation probability for each request in the CHATS instances and the random forest prediction model has shown to be the most effective.

Chapter 4

The Senior Transportation Problem

In this chapter, we formally define the senior transportation problem (STP) and state the specific characteristics that distinguish it from other transportation problems. Four exact methods including Mixed Integer Programming (MIP), Constraint Programming (CP), and two Logic-based Benders Decomposition (LBBD) models and a construction heuristic to solve the STP are then presented along with experimental results for each method in subsequent sections. Furthermore, the best approaches are tested on real-world instances. The chapter closes with empirical analysis of the best performing algorithms.

4.1 Problem Definition

The STP is a deterministic optimization problem in which a fixed fleet of heterogeneous vehicles from multiple depots must satisfy as many door-to-door transportation requests as possible within a fixed time horizon. All requests comprise of a one-to-one Pickup and Delivery Problem with Time Windows (PDPTW). Moreover, as the clients are seniors, a maximum ride time is enforced to minimize user inconvenience. With these characteristics, this problem is similar to the classical Dial-a-Ride Problem (DARP) or the PDPTW. However, due to the limited resources, not all requests can be met within the given horizon and therefore, the problem is to select a subset of requests such that the total weight of all served request is maximized. Additionally, part of the drivers may operate on a volunteer basis, thus aspects such as multi-depots, heterogeneous vehicles, and time windows on vehicles need to be considered.

The STP is described as follows. Let $G = (\mathcal{V}, \mathcal{A})$ be a directed complete graph with vertex set $\mathcal{V} = \mathcal{D} \cup \mathcal{N}$ where $\mathcal{D}$ represents the depot vertices and $\mathcal{N}$ represents the client vertices. A vertex represents a single geographic location, however, two vertices may share the same geographic location, and the travel time between these two vertices is 0. The set $\mathcal{D}$ is divided into $\mathcal{D}^+ = \{1, ..., |\mathcal{K}|\}$ (starting depot vertices) and $\mathcal{D}^- = \{|\mathcal{K}| + 1, ..., 2|\mathcal{K}|\}$ (ending depot vertices) where $|\mathcal{K}|$ is the number of vehicles. The set $\mathcal{N}$ is partitioned into $\mathcal{N}^+ = \{2|\mathcal{K}|+1, ..., 2|\mathcal{K}|+|\mathcal{R}|\}$ (pickup vertices) and $\mathcal{N}^- = \{2|\mathcal{K}|+|\mathcal{R}|+1, ..., 2|\mathcal{K}|+2|\mathcal{R}|\}$ (delivery vertices) where $|\mathcal{R}|$ is the number of requests. Each vertex $i \in \mathcal{V}$ is associated with a time window $[E_i, L_i]$ and a service duration S_i (how much time needs to be spent at the location). Each arc $(i, j) \in \mathcal{A}$ has a non-negative routing time $T_{i,j}$ satisfying the triangular inequality.

Let $\mathcal{K} = \{1, ..., |\mathcal{K}|\}$ represent the set of vehicles. Each vehicle $k \in \mathcal{K}$ is associated with a starting

depot $i_{k^+} \in \mathcal{D}^+$ and an ending depot $i_{k^-} \in \mathcal{D}^-$. Each vehicle must start and end at its associated depots. Multiple vehicles can share the same geographical location for its depots. However, relocation of vehicles between depots is not allowed. Each vehicle also specifies its availability. If a vehicle has multiple non-overlapping time windows, then it is considered as multiple vehicles. Furthermore, vehicles are heterogeneous and differ in capacity, thus each vehicle $k \in \mathcal{K}$ is associated with a maximum capacity C_k. Therefore, each vehicle has a known depot locations i_{k^+} and i_{k^-} with $S_{i_{k^+}} = S_{i_{k^-}} = 0$, a capacity C_k, and time windows $[E_{i_{k^+}}, L_{i_{k^+}}]$ and $[E_{i_{k^-}}, L_{i_{k^-}}]$ in which the vehicle must leave the starting depot, perform all pickup and delivery requests assigned to it and return to the ending depot.

Let $\mathcal{R} = \{1, ..., |\mathcal{R}|\}$ represent the set of requests. Each request r is paired with a positive weight, W_r denoting the importance of the request. The total weight of served requests contributes to the objective function. A request $r \in \mathcal{R}$ has an associated pickup location $i_{r^+} \in \mathcal{N}^+$ and a delivery location $i_{r^-} \in \mathcal{N}^-$. Requests are divided into two categories: *outbound* and *inbound* trips. In an outbound trip, the client is typically travelling from his/her home location to a destination location, and in an inbound trip, the client requests a return trip to their home location. In the context of the STP, the client only imposes a time window on the delivery location of an outbound trip and on the pickup location of an inbound trip. In addition, all clients are restricted to a maximum ride time, F, on any vehicle. Let Z be the end of the time horizon. Then for an outbound trip i, the time window associated with its pickup location is $[0, Z]$, whereas the time window of an inbound trip's delivery location is $[0, Z]$. The load size is positive for a pickup location vertex and negative for a delivery vertex, $Q_r = -Q_{|\mathcal{R}|+r}, \forall r \in \mathcal{R}^+$. A list of all parameters can be found in Table 4.1.

Table 4.1: STP parameters.

Parameter	Description
G	Graph representing the locations of vehicle depots and clients
$\mathcal{V}$	Set of vertices
$\mathcal{A}$	Set of arcs
$T_{i,j}$	Travel time between two location vertices in $\mathcal{V}$. All entries are strictly non-negative. A large entry indicates that no route is allowed between two locations. $T_{i,i} = 0, \forall i \in \mathcal{V}$ and $T_{i,j}$ does not have to equal $T_{j,i}$, tt represents the travel time matrix
$\mathcal{D}$	Set of depot location vertices, $\mathcal{D} \subset \mathcal{V}$
$\mathcal{K}$	Set of vehicle
$\|\mathcal{K}\|$	Number of available vehicles
$\mathcal{N}$	Set of client location vertices, $\mathcal{N} \subset \mathcal{V}$
$\mathcal{R}$	Set of requests
$\|\mathcal{R}\|$	Number of requests
E_i	Earliest service start time at each location i in $\mathcal{V}$
L_i	Latest service start time at each location i in $\mathcal{V}$
S_i	Service time needed to perform the task at each location i in $\mathcal{V}$

Q_r	Load size associated with each request r in $\mathcal{R}$
W_r	Weight associated with each request r in $\mathcal{R}$. All entries are strictly non-negative.
C_k	Capacity of each vehicle k in $\mathcal{K}$
Z	Maximum time horizon
F	Maximum ride time of a client

A *route* for vehicle k is a sequence of vertices, $[i_{k^+}, ..., i_{k^-}]$. A request is *served* when it is part of a route. The set of routes must satisfy the following constraints:

1. The pickup and delivery vertices of any request must be on the same route;
2. The pickup vertex must precede the delivery vertex;
3. A vertex is visited by at most one vehicle;
4. The load of a vehicle k cannot exceed its maximum capacity C_k at any point;
5. A route must start and end within the vehicle's availability window;
6. No subtours are allowed in any route;
7. The ride time of a client cannot exceed the maximum ride time F;
8. All pickup and delivery must be served within their specified time windows.

4.1.1 Problem Complexity

Each request r is given a weight W_r, each vehicle k is given a capacity C_k, and the objective of the STP is to maximize the total weight of all served requests that can fit into the given vehicles and satisfying the route constraints. The objective function can be modelled as Equation 4.1 where $\varphi_{k,r}$ is 1 if request r is assigned to vehicle k and 0 otherwise.

$$\text{maximize} \sum_{k \in \mathcal{K}} \sum_{r \in \mathcal{R}} (W_r \times \varphi_{k,r}) \tag{4.1}$$

The STP's objective function is equivalent to the one in the Multiple Knapsack Problem (MKP) as shown in Equation 4.2 where p_j is the profit of item j, equivalent to W_r in STP and $x_{i,j}$ is 1 if item j is assigned to knapsack i and 0 otherwise. The variable $x_{i,j}$ is equivalent to the the variable $\varphi_{k,r}$ in the STP.

$$\text{maximize} \sum_{i=1}^{m} \sum_{j=1}^{n} (p_j \times x_{i,j}) \tag{4.2}$$

Furthermore, the STP derives some of its constraints such as constant flow, capacity and time window constraints from the PDPTW and the rest of the constraints such as selectivity of requests and the maximum time horizon are explicit to the STP. Therefore, the STP can be generalized to an MKP with constraints similar to those in the PDPTW and additional constraints specific to the STP. Since

both the MKP and the PDPTW are $\mathcal{NP}$-hard problems [28, 45], the STP, as a combination of the two, is also $\mathcal{NP}$-hard.

4.2 Solution Approaches for the STP

Four exact methods, one MIP, one CP, and two LBBD approaches and one heuristic have been developed to solve the STP. This section presents the variables used and the mathematical formulations of each of the five methods.

4.2.1 A MIP Approach

A pure MIP model for the STP is presented in this section. The model is an extension of Ropke et al.'s MIP formulation [57] that is the widely used base model for the PDPTW.

Decision Variables

This formulation uses three variables: a binary variable $x_{k,i,j}$ and two continuous variables $v_{k,i}$ and $u_{k,i}$. $x_{k,i,j} = 1$ if vehicle k visits location j immediately after visiting location i and 0 otherwise. $x_{k,i,j}$ is only instantiated for $(i,j) \in \mathcal{A}$. $(i,j) \in \mathcal{A}$ if one of the following is true: 1. $i \in \mathcal{D}^+$ and $j \in \mathcal{N}^+$, 2. both i and $j \in \mathcal{N}$, 3. $i \in \mathcal{N}^-$ and $j \in \mathcal{D}^-$, or 4. $i \in \mathcal{N}^+$ and $j \in \mathcal{N}^-$. $v_{k,i}$ is a continuous variable that indicates the load of the vehicle k after visiting location $i \in \mathcal{V}$. It is strictly positive and less than or equal to the maximal vehicle capacity $\max\{C_k\}, \forall k \in \mathcal{K}$. $u_{k,i}$ is a continuous variable that indicates the time when vehicle k leaves location $i \in \mathcal{V}$. It is strictly positive and is less than or equal to the maximum time horizon Z. A summary of the variables is given in Table 4.2.

Table 4.2: Variables for MIP model.

Location Variables	**Type**	**Description**
$x_{k,i,j}$	binary	1 if arc $(i,j) \in \mathcal{A}$ is used by vehicle k in the final solution and 0 otherwise.
Load Variables	**Type**	**Description**
$v_{k,i}$	continuous	Load of vehicle $k \in \mathcal{K}$ after visiting node $i \in \mathcal{V}$
Time Windows Variables	**Type**	**Description**
$u_{k,i}$	continuous	The time when vehicle $k \in \mathcal{K}$ leaves node $i \in \mathcal{V}$ after completing the service at i

MIP Formulation

The formulation is given in (4.3) - (4.17).

$$\max \sum_{k\in\mathcal{K}}\sum_{r\in\mathcal{R}}\sum_{j\in\mathcal{N}}\left(W_r \times x_{k,i_{r^+},j}\right) \tag{4.3}$$

$$\text{s.t.} \sum_{j\in\mathcal{N}^+} x_{k,i_{k^+},j} + x_{k,i_{k^+},i_{k^-}} = 1 \qquad \forall k \in \mathcal{K} \tag{4.4}$$

$$\sum_{i\in\mathcal{N}^-} x_{k,i,i_{k^-}} + x_{k,i_{k^+},i_{k^-}} = 1 \qquad \forall k \in \mathcal{K} \tag{4.5}$$

$$\sum_{k\in\mathcal{K}}\sum_{j\in\mathcal{V}} x_{k,i_{r^+},j} \leq 1 \qquad \forall r \in \mathcal{R} \tag{4.6}$$

$$\sum_{j\in\mathcal{V}} \left(x_{k,i,j} - x_{k,j,i}\right) = 0 \qquad \forall k \in \mathcal{K}, i \in \mathcal{N} \tag{4.7}$$

$$\sum_{j\in\mathcal{V}} \left(x_{k,i_{r^+},j} - x_{k,j,i_{r^-}}\right) = 0 \qquad \forall k \in \mathcal{K}, r \in \mathcal{R} \tag{4.8}$$

$$u_{k,j} \geq (u_{k,i} + T_{i,j} + S_j) - M \times (1 - x_{k,i,j}) \qquad \forall k \in \mathcal{K}, i, j \in \mathcal{V} \tag{4.9}$$

$$u_{k,i} \geq E_i - M \times \left(1 - \sum_{j\in\mathcal{V}} x_{k,i,j}\right) \qquad \forall k \in \mathcal{K}, i \in \mathcal{V} \tag{4.10}$$

$$u_{k,i} \leq L_i - S_i + M \times \left(1 - \sum_{j\in\mathcal{V}} x_{k,i,j}\right) \qquad \forall k \in \mathcal{K}, i \in \mathcal{V} \tag{4.11}$$

$$u_{k,i_{r^+}} \leq u_{k,i_{r^-}} \qquad \forall k \in \mathcal{K}, r \in \mathcal{R} \tag{4.12}$$

$$\left(u_{k,i_{r^-}} - u_{k,i_{r^+}}\right) \leq F \qquad \forall k \in \mathcal{K}, r \in \mathcal{R} \tag{4.13}$$

$$v_{k,j} \geq (v_{k,i} + Q_i) - M \times (1 - x_{k,i,j}) \qquad \forall k \in \mathcal{K}, i, j \in \mathcal{V} \tag{4.14}$$

$$x_{k,i,j} \in \{0, 1\} \qquad \forall k \in \mathcal{K}, (i, j) \in \mathcal{A} \tag{4.15}$$

$$0 \leq u_{k,i} \leq Z \qquad \forall k \in \mathcal{K}, i \in \mathcal{V} \tag{4.16}$$

$$0 \leq v_{k,i} \leq C_k \qquad \forall k \in \mathcal{K}, i \in \mathcal{V} \tag{4.17}$$

The objective function (4.3) maximizes the sum of weights of served requests. Constraints (4.4) and (4.5) ensure that each vehicle leaves from its starting depot and ends at its ending depot. Constraints (4.6) allow for the selectivity of requests. Constant flow is enforced with Constraints (4.7). Both the pickup and delivery locations must be visited by the same vehicle as enforced through Constraints (4.8). In Constraints (4.9), the travel time and service time of visited nodes are enforced. Constraints (4.10) and (4.11) make sure that each node that is visited must be visited within its time window. Constraints (4.12) impose that pickup nodes must precede delivery nodes. Constraints (4.13) enforce that each ride does not exceed the maximum ride time. The capacity Constraints (4.14) keep track of the load of each vehicle after visiting the node. Constraints (4.15), (4.16) and (4.17) bound the variables x, u and v, respectively.

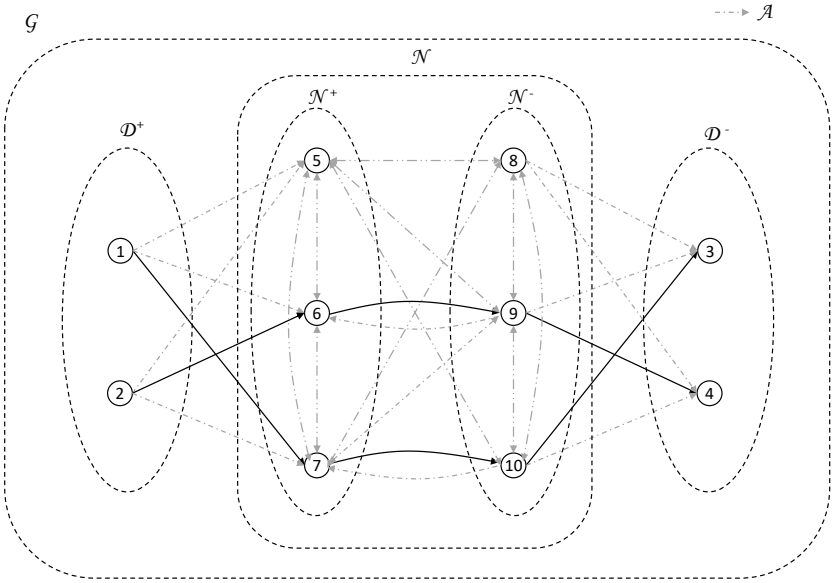
Compared to Rokpe et al.'s model [57], the objective function has been modified to reflect the selective nature of the STP. Constraints (4.4) and (4.5) have been added to model the multi-depot aspect and that not all vehicles have to be used in the final solution. The equal sign has been changed to a $\leq$ in Constraints (4.6) in order to model the selectivity. A big M is introduced in Constraints (4.9), (4.10),

(4.11), and (4.14) to model the optionality of location visits. Constraints (4.13) are a new constraint to represent the maximum ride time constraint.

Model Size

There are $|\mathcal{K}|$ vehicles, $|\mathcal{R}|$ requests and $|\mathcal{V}| = 2(|\mathcal{K}| + |\mathcal{R}|)$ vertices. Then, the cardinality of the set $\mathcal{A}$ is $2(|\mathcal{K}| \times |\mathcal{R}|) + (2|\mathcal{R}| - 1)^2$ where $2(|\mathcal{K}| \times |\mathcal{R}|)$ is the number of arcs (i, j) such that $i \in \mathcal{D}^+$ and $j \in \mathcal{N}^+$ or $i \in \mathcal{N}^-$ and $j \in \mathcal{D}^-$, while $(2|\mathcal{R}| - 1)^2$ is the number of arcs (i, j) such that $i \in \mathcal{N}$ and $j \in \mathcal{N}, i \neq j$. Thus, $|\mathcal{K}| \left(2(|\mathcal{K}| \times |\mathcal{R}|) + (2|\mathcal{R}| - 1)^2\right) = 2|\mathcal{K}|^2|\mathcal{R}| + (2|\mathcal{R}| - 1)^2$ number of $x_{k,i,j}$ variables will be generated along with $(|\mathcal{K}| \times |\mathcal{V}|) = 2|\mathcal{K}|^2 + 2(|\mathcal{K}| \times |\mathcal{R}|)$ number of $v_{k,i}$ and $u_{k,i}$ variables. Thus this algorithm requires $O(|\mathcal{K}|^2|\mathcal{R}| + |\mathcal{R}|^2)$ number of variables.

Figure 4.1: Example of an STP graph.



In the example shown in Figure 4.1, there are $|\mathcal{K}| = 2$ vehicles, $|\mathcal{R}| = 3$ requests and $|\mathcal{V}| = 10$ vertices, thus $\left(2 \times (2 \times 3) + (6 - 1)^2\right) = 42$ $x_{k,i,j}$ variables and 20 $v_{k,i}$ and $u_{k,i}$ variables.

4.2.2 A CP Approach

A CP Model solving the STP is presented in this section. In this model, each location is represented as an interval variable where the variable domain is its possible time windows. Each location variable is then assigned to a vehicle and corresponding route while satisfying capacity and ride time constraints. The use of interval variables makes the representation of the time windows clear and concise.

Decision Variables

This formulation uses only interval variables and uses cumulative functions and sequence expressions to logically link the variables. Each location $i \in \mathcal{V}$ is an optional interval variable x_i that is bounded by its

time window and the length of its service time. We assume that each vehicle visits its depot locations regardless if it is assigned requests or not. The presence of x_i in the final solution implies that the location is visited by a vehicle. Auxiliary interval variables $X_{i,k}$ and $\bar{X}_{k,i}$ logically link the x_i variables to vehicles. These variables are again optional and the presence of $X_{i,k}$ and $\bar{X}_{k,i}$ indicates that location i is visited by vehicle k. Cumulative functions $v_{k,i}$ are expressions that model the load of vehicle k after visiting location i. Finally, each route is modelled by a sequence variable u_k that is just a permutation of locations visited by vehicle k. A summary of the variables is given in Table 4.3.

Table 4.3: Variables for Vehicle-Based Model.

Location Variables	**Type**	**Description**
x_i	interval	Time interval in which location $i \in \mathcal{V}$ is served and not present if the location is not served. The variable is bounded by the time window at location i
$X_{i,k}$	interval	Time interval in which location $i \in \mathcal{N}$ is visited by vehicle $k \in \mathcal{K}$ and not present if location i is not visited by vehicle k. ($\mathbf{X_i}$ represents a vector of the $X_{i,k}$ variables
$\bar{X}_{k,i}$	interval	Time interval in which vehicle $k \in \mathcal{K}$ visits location $i \in \mathcal{N}$
Load Variables	**Type**	**Description**
$v_{k,i}$	cumul function	Load of vehicle $k \in \mathcal{K}$ after visiting node $i \in \mathcal{N}$
Sequence Variables	**Type**	**Description**
u_k	sequence	Sequence of locations visited by vehicle $k \in \mathcal{K}$

CP Formulation

The CP formulation is presented in Equations (4.18) - (4.27).

$$\text{maximize} \sum_{r \in \mathcal{R}} \left(W_r \times \texttt{PresenceOf}(x_{i_{r^+}}) \right) \tag{4.18}$$

subject to

Location Constraints

$$\texttt{Alternative}(x_i, \mathbf{X_i}) \qquad \forall i \in \mathcal{N} \tag{4.19}$$

$$\texttt{Before}(\bar{X}_{k,i_{r^+}}, \bar{X}_{k,i_{r^-}}) \qquad \forall k \in \mathcal{K}, \forall r \in \mathcal{R} \tag{4.20}$$

$$\texttt{PresenceOf}(\bar{X}_{k,i_{r^+}}) = \texttt{PresenceOf}(\bar{X}_{k,i_{r^-}}) \qquad \forall k \in \mathcal{K}, \forall r \in \mathcal{R} \tag{4.21}$$

Ride Time Constraint

$$\texttt{GetStartMax}(x_{i_{r^-}}) - \texttt{GetEndMax}(x_{i_{r^+}}) \leq F \qquad \forall r \in \mathcal{R} \tag{4.22}$$

Capacity Constraints

$$v_{k,i} = \texttt{StepAtStart}(\bar{X}_{k,i}, Q_i) \qquad \forall k \in \mathcal{K}, \forall i \in \mathcal{N} \tag{4.23}$$

$$\sum_{i \in \mathcal{N}} v_{k,i} \leq C_k \qquad \forall k \in \mathcal{K} \tag{4.24}$$

Route Constraints

$$\texttt{First}(u_k, x_{i_{k^+}}) \qquad \forall k \in \mathcal{K} \tag{4.25}$$

$$\texttt{Last}(u_k, x_{i_{k^-}}) \qquad \forall k \in \mathcal{K} \tag{4.26}$$

$$\texttt{NoOverlap}(u_k, tt) \qquad \forall k \in \mathcal{K} \tag{4.27}$$

The objective function (4.18) chooses a set of locations to visit in order to maximize the total weight of requests fulfilled while respecting the constraints. The function `PresenceOf` indicates if the variable x_i is present or not (i.e., if the location is served or not) in a solution.

Constraints (4.19) make sure that if location i is visited, then only one vehicle serves that location. The sequence order for pickup and delivery locations is ensured by Constrains (4.20). Constraints (4.21) enforces that if the pickup location of a request is served by vehicle k, then its associated delivery location must also be served by the same vehicle k. The maximum ride time constraint is modelled through Constraints (4.22).

Constraints (4.23) use the cumulative function to keep track of the load size of the vehicles after visiting location i. Constraints (4.24) enforces that the capacity of the vehicle must not be exceeded.

Furthermore, Constraints (4.25) and (4.26) indicate that each route must start at its associated start depot and end at its associated ending depot locations. The CP model uses the `NoOverlap` global Constraints (4.27) to enforce subtour elimination on each route.

Model Size

Compared to the MIP model, the number of variables and the number of constraints in the CP model are significantly reduced. Since there is no use of arc variables, this CP model only requires $O(k|\mathcal{R}|)$ variables. More details on the number of variables and constraints on larger instances for the CP model are presented in Section 3.2.

4.2.3 Logic-based Benders Decomposition Approaches

Two logic-based Benders decomposition (LBBD) [34] approaches to solve the STP are presented in this section. We break the STP into a relaxed master problem and a number of subproblems. The relaxed master problem is a packing and assignment problem and each subproblem is an optimal routing and scheduling problem. We present two MIP formulations of the master problem, one more restricted and one more relaxed, as well as a CP formulation. The subproblem is modelled using a CP formulation. The master problem finds the optimal assignment of requests to vehicles given the different relaxations, then several subproblems are created. Each subproblem is an optimization problem to find the optimal route given the assigned requests. If the optimal route for each subproblem satisfies all requests assigned to it, then the optimal solution has been found, otherwise, a Benders cut is produced for every subproblem that cannot meet all requests. This LBBD model finds a feasible global solution at every iteration because each the route found in each subproblem is feasible. The LBBD framework for the STP is represented in Figure 4.2.

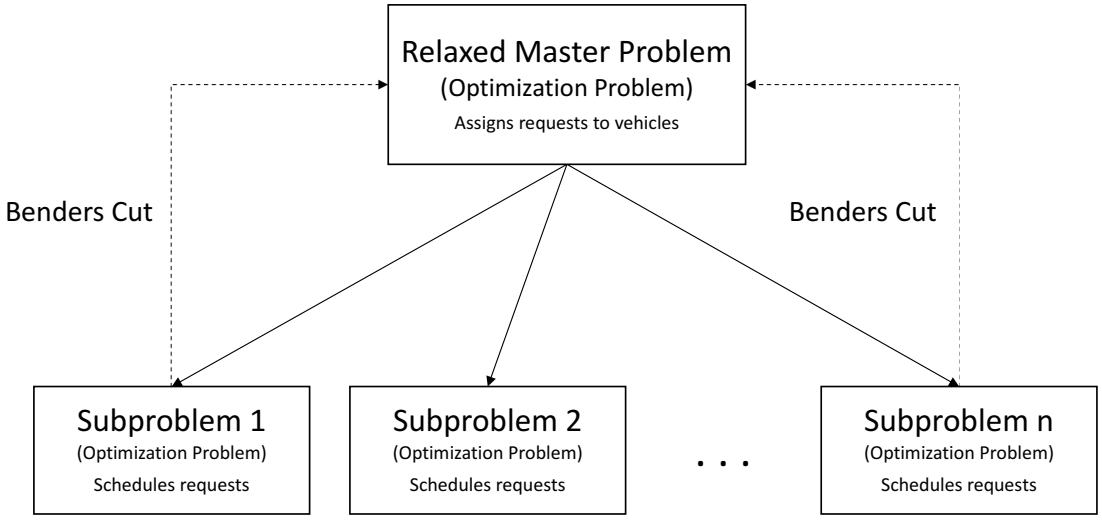


Figure 4.2: The LBBD framework for the STP.

MIP Master Problem

The master problem makes the decision of "packing" each request into a vehicle given the vehicle's time window and capacity using integer decision variables $\varphi_{k,r}$ which equal to 1 if request r is assigned to vehicle k and 0 otherwise. There are three relaxations in the master problem. In the first relaxation, the variable $x_{k,i,j}$ determining if a specific job j is visited directly after job i is relaxed to be continuous. Thus, instead of a single sequence of locations, the service time of each location is allowed to be divided into small partitions. This relaxation is similar to the time-index model of a scheduling problem as described in [64].

Secondly, the sequence constraint between pickup and delivery locations is relaxed; the pickup location does not need to be visited before the delivery location. The reasoning about sequences are

described using decision variables $y_{k,i}$ which is equal to 1 if vehicle k visits location i and 0 otherwise.

In the third relaxation, similar to the relaxations described in [32], a set of area relaxations based on the size and the minimum time required of each request is implemented. Each request is represented as a "job" to schedule. The size of the request r, Q_r serves as the "height" of the job. The minimum time required, defined as the sum of the service time of the pickup location i_{r^+}, the service time of the delivery location i_{r^-} and the travel time between the two locations, serves as the "width" of the job. Thus for each request $r \in \mathcal{R}$, $\zeta_r = S_{i_{r^+}} + T_{i_{r^+},i_{r^-}} + S_{i_{r^-}}$ is constrained to occur within a time window $[E_{i_{r^+}}, L_{i_{r^-}}]$ where $E_{i_{r^+}}$ is the earliest start time of the pickup location and $L_{i_{r^-}}$ is the latest finish time of the delivery location. Figure 4.3 shows an example of a request.

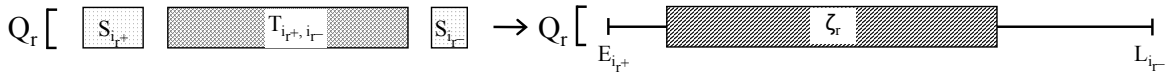


Figure 4.3: Example of a request.

Let $\mathcal{W} = \{(t_1, t_2) | t_1 \in \mathcal{E}, t_2 \in \mathcal{L}, t_1 < t_2\}$ be set of time windows where $\mathcal{E}$ is the set of all earliest start times E_r and $\mathcal{L}$ is the set of all latest finish times L_r. Let $\mathcal{R}(t_1, t_2) = \{r \in \mathcal{R} | t_1 \leq E_r, t_2 \geq L_r\}$ be the set of all requests that can be executed between t_1 and t_2. The area relaxations consist of a set of linear constraints as below.

$$\sum_{r \in \mathcal{R}(t_1,t_2)} \zeta_r \times Q_r \times \varphi_{k,r} \leq C_k \times (t_2 - t_1) \qquad \forall k \in \mathcal{K}, (t_1, t_2) \in \mathcal{W}$$

Since the master problem has been relaxed to exclude specific time windows and capacity constraints, the master problem only makes use of three location decision variables. $x_{k,i,j}$ variables, similar to those of the MIP model in Section 4.2.1, have been relaxed to be continuous, binary $y_{k,i}$ variables are associated with the relaxed sequence constraints and binary $\varphi_{k,r}$ variables are associated with the relaxed area packing constraints. $x_{k,i,j}$ must be within $[0, 1]$ and indicates the percentage of location j directly after location i on vehicle k. $y_{k,i} = 1$ if location i is visited by vehicle k and 0 otherwise. Furthermore, $\varphi_{k,r} = 1$ if request r is served by vehicle k and 0 otherwise. All variables are summarized in Table 4.4. The formulation for the MIP master problem is presented in Equations (4.28) - (4.41).

The number of variables required for the master problem is similar to the pure MIP model due to the presence of $x_{k,i,j}$ variables and is of order $O(k^2 r + k r^2)$ with the change that $x_{k,i,j}$ are now continuous variables and a number of constraints have been removed and replaced.

Table 4.4: Variables for master MIP model.

Variables	**Type**	**Description**
$x_{k,i,j}$	continuous	[0,1] indicating the part of location $j \in V$ that is visited directly after location $i \in V$ by vehicle k and 0 if no part of location j is visited directly after location i.
$y_{k,i}$	binary	1 if location $i \in \mathcal{N}$ is visited by vehicle $k \in \mathcal{K}$ and 0 otherwise.
$\varphi_{k,r}$	binary	1 if request $r \in \mathcal{R}$ is served by vehicle $k \in \mathcal{K}$ and 0 otherwise.

$$\text{maximize} \sum_{r \in \mathcal{R}} \sum_{k \in \mathcal{K}} (W_r \times \varphi_{k,r}) \tag{4.28}$$

$$\text{subject to} \sum_{k \in \mathcal{K}} y_{k,i} \leq 1 \qquad \forall i \in \mathcal{N} \tag{4.29}$$

$$y_{k,i} = \sum_{j \in \mathcal{N}} x_{k,i,j} \qquad \forall k \in \mathcal{K}, i \in \mathcal{N} \tag{4.30}$$

$$y_{k,j} = \sum_{i \in \mathcal{N}} x_{k,i,j} \qquad \forall k \in \mathcal{K}, j \in \mathcal{N} \tag{4.31}$$

$$Q_r \times \varphi_{k,r} \leq C_k \qquad \forall k \in \mathcal{K}, r \in \mathcal{R} \tag{4.32}$$

$$\zeta_r = S_{i_{r^+}} + T_{i_{r^+}, i_{r^-}} + S_{i_{r^-}} \qquad \forall r \in \mathcal{R} \tag{4.33}$$

$$(E_{i_{r^+}} + \zeta_r) \times \varphi_{k,r} \leq L_{i_{k^-}} \qquad \forall k \in \mathcal{K}, r \in \mathcal{R} \tag{4.34}$$

$$L_{i_{r^-}} - \zeta_r \geq E_{i_{k^+}} \times \varphi_{k,r} \qquad \forall k \in \mathcal{K}, r \in \mathcal{R} \tag{4.35}$$

$$\sum_{i \in \mathcal{N}} (S_i \times y_{k,i}) + \xi_k \leq L_k - E_k \qquad \forall k \in \mathcal{K} \tag{4.36}$$

$$\xi_k = \sum_{(i,j) \in \mathcal{A}} x_{k,i,j} \times T_{i,j} \qquad \forall k \in \mathcal{K} \tag{4.37}$$

$$\sum_{r \in \mathcal{R}(t_1,t_2)} \zeta_r \times Q_r \times \varphi_{k,r} \leq C_k \times (t_2 - t_1) \qquad \forall k \in \mathcal{K}, (t_1, t_2) \in \mathcal{W} \tag{4.38}$$

$$y_{k,r} = y_{k,r+|\mathcal{R}|} = \varphi_{k,r} \qquad \forall k \in \mathcal{K}, r \in \mathcal{R} \tag{4.39}$$

$$y_{k,i}, \varphi_{k,r} \in \{0, 1\} \qquad \forall k \in \mathcal{K}, i \in \mathcal{N}, r \in \mathcal{R} \tag{4.40}$$

$$x_{k,i,j} \in [0, 1] \qquad \forall k \in \mathcal{K}, i, j \in \mathcal{N} \tag{4.41}$$

Benders Cuts

The objective function (4.28) maximizes the total weight of all the requests served. Constraints (4.29) makes sure all locations are visited at most once. Constraints (4.30) and (4.31) ensure the consistency between the y variables and the x variables. The approximate length of a request, ζ_r is modelled in Constraints (4.33) and Constraints (4.32), (4.34), and (4.35) remove all infeasible requests from a specific vehicle. The sum of service time and relaxed travel time is bounded by each vehicle's time window in Constraints (4.36) and the relaxed travel time for each vehicle is defined in Constraints (4.37). Constraints (4.38) represents the area relaxations. The relationship between the $y_{k,i}$ and $\varphi_{k,r}$ variables is established in Constraints (4.39). It also specifies that if the pickup location of a request is assigned to a vehicle then the delivery location must be visited by the same vehicle. Constraints (4.40) and (4.41) form the bounds of the variables.

Simplifying the MIP Master Problem

When solving large real-world problem instances, this MIP formulation for the master problem fails to build the model as the number of variables and constraints becomes too big and the computer runs out of memory. In the single largest instance, the number of variables is 2.52 million and the number of constraints is 2.24 million. In order to reduce the memory usage, we removed all the $x_{k,i,j}$ variables along with their related constraints, namely Constraints (4.30) and (4.31). The distance constraints

(4.36) and (4.37) are replaced with a single simpler constraint. Instead of modelling the exact travel distance between consecutive locations, we compute the minimum travel time from each location i, denoted with $\mathcal{T}_i$, and use that as our relaxed distance constraint as shown in (4.42).

$$\sum_{i \in \mathcal{N}} (y_{k,i} \times \mathcal{T}_i + S_i) + \mathcal{T}_{i_{k^+}} + S_{i_{k^+}} \leq L_{i_{k^-}} - E_{i_{k^+}} \qquad \forall k \in \mathcal{K} \qquad (4.42)$$

The area relaxations (4.38) are also removed as these generate $O((|\mathcal{K}| \times |\mathcal{R}|)^2)$ constraints. The new MIP master problem is as below.

maximize Objective Function (4.28)

subject to Constraints (4.29), (4.32), (4.33), (4.34), (4.35)

$$\sum_{i \in \mathcal{N}} (y_{k,i} \times \mathcal{T}_i + S_i) + \mathcal{T}_{i_{k^+}} + S_{i_{k^+}} \leq L_{i_{k^-}} - E_{i_{k^+}} \qquad \forall k \in \mathcal{K} \qquad (4.42)$$

Constraints (4.39), (4.40)

Benders Cuts

All results reported in this thesis use this simplified master problem.

CP Master Problem

In another attempt to decrease the size of the master problem, we provide a CP formulation that uses a significantly fewer number of variables. Since we are relaxing all the temporal constraints in the master problem, there is no need for sequence variables. In this CP formulation of the LBBD master problem, we only employ interval variables x_i and $X_{i,k}$ as defined in Section 4.2.2. The formulation for the CP master problem is given in Equations (4.43) - (4.49).

$$\text{maximize} \sum_{r \in \mathcal{R}} \left(W_r \times \texttt{PresenceOf}(x_{i_{r^+}}) \right) \tag{4.43}$$

subject to

Location Constraints

$$\texttt{Alternative}(x_i, \mathbf{X}_i) \qquad \forall i \in \mathcal{N} \tag{4.44}$$

$$\texttt{PresenceOf}(X_{i_{r^+},k}) = \texttt{PresenceOf}(X_{i_{r^-},k}) \qquad \forall k \in \mathcal{K}, r \in \mathcal{R} \tag{4.45}$$

Route Constraints

$$\texttt{EndBeforeStart}(x_{i_{k^+}}, X_{i,k}) \qquad \forall k \in \mathcal{K}, i \in \mathcal{N} \tag{4.46}$$

$$\texttt{EndBeforeStart}(X_{i,k}, x_{i_{k^-}}) \qquad \forall k \in \mathcal{K}, i \in \mathcal{N} \tag{4.47}$$

$$\texttt{EndBeforeStart}(x_{i_{k^+}}, x_{i_{k^-}}) \qquad \forall k \in \mathcal{K} \tag{4.48}$$

Distance Constraint

$$\sum_{i \in \mathcal{N}} (\texttt{PresenceOf}(X_{i,k}) \times \mathcal{T}_i + S_i) + \mathcal{T}_{i_{k^+}} + S_{i_{k^+}} \leq L_{i_{k^-}} - E_{i_{k^+}} \qquad \forall k \in \mathcal{K} \tag{4.49}$$

Benders Cuts

The objective function (4.43) remains the same along with Constraints (4.44) and (4.45). Since sequences are relaxed, no sequence variables are modelled but Constraints (4.46), (4.47), and (4.48) still make sure that each vehicle visits its starting depot before visiting any other location and visits its ending depot location last. Finally, the distance relaxation is the same as in the MIP master problem represented by Constraints (4.49).

CP Subproblem

After the master problem assigns requests to vehicles, a subproblem is created for each vehicle that has been assigned at least two requests. Each subproblem is a single vehicle STP that maximizes the total weight of served requests given the attributes of the vehicle and the requests assigned by the master problem. If the subproblem is able to schedule all the requests given to it, then the vehicle has a feasible assignment. Otherwise, the vehicle is not feasible and the solution of the subproblem is the optimal assignment for that specific subset of requests. The objective value of the subproblem is then returned to the master problem as a Benders cut. With optimization subproblems, at each iteration of the LBBD, the algorithm finds a globally feasible solution. The subproblem is modelled using CP and the formulation is similar to that of the model provided in Section 4.2.2 with the removal of Constraints (4.19) and (4.21) since each subproblem is a single vehicle problem. Let k^* represent the vehicle, $\mathcal{R}^*$ the subset of requests assigned to k^* from the master problem, $\mathcal{N}^*$ the set of clients and $\mathcal{V}^*$ the set of vertices that forms the graph in the subproblem.

Since each subproblem schedules one vehicle, the variables $X_{i,k}$ and $\bar{X}_{k,i}$ are no longer needed. The CP formulation of the subproblem uses three decision variables. For each location $i \in \mathcal{V}^*$, the optional interval variable x_i represents the time interval in which location i is served and is not present if it is not visited. This variable is bounded by the time window of the specific location. Cumulative functions y_i represent the load of the vehicle after visiting location i. Finally a sequence variable u represents the sequence of visits of the vehicle. All variables are summarized in Table 4.5. The formulation for the CP

subproblem is presented in Equations (4.50) - (4.57).

Location Variables	**Type**	**Description**
x_i	interval	Time interval in which location $i \in \mathcal{V}^*$ is served and is not present if the location is not visited. The variable is bounded by the time window at location i
Load Variables	**Type**	**Description**
v_i	cumul function	Load of vehicle after visiting location $i \in \mathcal{N}^*$
Sequence Variable	**Type**	**Description**
u	sequence	Sequence of locations visited by the vehicle

Table 4.5: Variables for the CP formulation of LBBD's subproblem.

$$\text{maximize} \sum_{r \in \mathcal{R}^*} \left(W_r \times \texttt{PresenceOf}(x_{i_{r+}}) \right) \tag{4.50}$$

subject to

Location Constraints

$$\texttt{Before}(u, x_{i^+}, x_{i^-}) \qquad \forall i \in \mathcal{R}^* \tag{4.51}$$

Ride Time Constraints

$$\texttt{GetStartMax}(x_{i^-}) - \texttt{GetEndMax}(x_{i^+}) \leq F \qquad \forall i \in \mathcal{R}^* \tag{4.52}$$

Capacity Constraints

$$v_i = \texttt{StepAtStart}(v_i, Q_i) \qquad \forall i \in \mathcal{N}^* \tag{4.53}$$

$$0 \leq \sum_{i \in \mathcal{N}^*} uvi \leq C_{k^*} \tag{4.54}$$

Route Constraints

$$\texttt{First}(u, x_{i_{k^*+}}) \tag{4.55}$$

$$\texttt{Last}(u, x_{i_{k^*-}}) \tag{4.56}$$

$$\texttt{NoOverlap}(u) \tag{4.57}$$

The subproblem is to find the maximal weight of served requests given a vehicle and a subset of requests, thus similar to the full CP model presented in Section 4.2.2. The objective function maximizes the sum of weights of served requests. Constraints (4.51) make sure that the pickup location is visited before the delivery location. The maximum ride time is enforced through Constraints (4.52). Constraints (4.53) and (4.54) use the step function in CP to keep track of the load of the vehicle after visiting each location and makes sure that the load does not exceed the capacity of the vehicle at any location. Constraint (4.55) forces the start of the sequence at the starting depot of the vehicle and Constraint (4.56) makes sure the sequence ends at the ending depot of the vehicle. Finally, Constraint (4.57) takes into account the travel distances between locations for the sequence and eliminates subtours.

Benders Cut

In each iteration h, if the subproblem schedules all the requests assigned to it, then this subproblem is feasible. Otherwise, an optimality cut is returned to the master problem, or in other words, a constraint is added to the model. The cut specifies that given the subset of requests $\mathcal{R}^*$ to vehicle k^*, denoted by $\mathcal{J}_{h,k}$, the objective value of this combination cannot be larger than the subproblem's optimal value denoted by z^*. This cut is modelled in a MIP formulation as in Inequality (4.58) and in a CP formulation as in Inequality (4.59). The optimality cut has been proved to be valid in previous work [58].

$$\sum_{r \in \mathcal{J}_{h,k}} (\varphi_{k,r} \times W_r) \leq z^* \qquad \forall k \in \mathcal{K}, h \in \{1, ..., H-1\} \tag{4.58}$$

$$\sum_{r \in \mathcal{J}_{h,k}} (\texttt{PresenceOf}(X_{i_{r^+},k} \times W_r) \leq z^* \qquad \forall k \in \mathcal{K}, h \in \{1, ..., H-1\} \tag{4.59}$$

4.2.4 A Heuristic Approach

In order to compare the different approaches to a runtime-efficient heuristic algorithm, we also developed a construction heuristic. Since the objective function is to maximize the total weight of the requests served, it is logical to try to schedule the requests that have the highest ratio of weight to length first. Furthermore, vehicles are sorted in ascending order of time availability length so that requests are spread out amongst all vehicles and not concentrated on a single vehicle with a large time window. The algorithm schedules the highest weight ratio request to the first vehicle that can perform the request. If no currently available vehicle can satisfy a request, then the request is not scheduled. The algorithm is

outlined in Algorithm 1.

Algorithm 1: Construction Heuristic for the STP.

Data: Set of requests $\mathcal{R}$, and set of vehicles $\mathcal{K}$
Result: A set of scheduled routes
Sort $\mathcal{R}$ based on descending order of $\frac{weight}{length}$;
Sort $\mathcal{K}$ based on ascending order time window size;
for *all requests* r *in* $\mathcal{R}$ **do**
  for *all vehicles* v *in* $\mathcal{K}$ **do**
    if r *can be served by* v **then**
      assign r to v and set start time of r as earliest start time on r that is after the earliest pickup time of r;
      split v into two vehicle pieces, v_1 and v_2;
      set start and end locations and start and end times for v_1 and v_2;
      insert v_1 and v_2 into $\mathcal{K}$ based on the new time window lengths;
      break;
    else
      move to the next vehicle;
    end
  end
end
Regroup all pieces of the same vehicle to make scheduled routes;

4.3 Experimental Results

In this section, we first discuss the experimental results of running the five different methodologies on the randomly generated instances. CP and MIP/CP LBBD show the best performance, thus these two methodologies are then tested on the CHATS dataset.

All five approaches are coded using IBM's CPLEX Studio 12.7 in C++. The experiments are run on a computer with Intel Xeon E3-1226 v3 @ 3.30GHz, 16G RAM using single thread and a 600 second runtime limit.

4.3.1 Randomly Generated Dataset Results

In descending order of best performance, CP solved all 75 (100%) instances to optimality in an average of 2.75 seconds, MIP/CP LBBD solved 71 (95%) instances with an average runtime of 25.13 seconds, CP/CP LBBD solved 49 (65%) instances with an average of 110.14 seconds and MIP solved 35 (48%) instances with an average of 90.00 seconds. Each of the four exact methods are then run with the Heuristic solution as a warm start. It can be seen in Figure 4.4 that MIP and CP/CP LBBD with the Heuristic start have a substantial improvement. However, all the instances that were solved to optimality with the Heuristic but failed to solve without are the ones where the heuristic found an optimal solution by itself. The runtime for both MIP and CP/CP LBBD have also improved significantly. The heuristic start only improves MIP/CP LBBD a little while it has very minimal effects on CP.

The runtime of MIP, CP/CP LBBD and MIP/CP LBBD are compared to that of CP in Figure

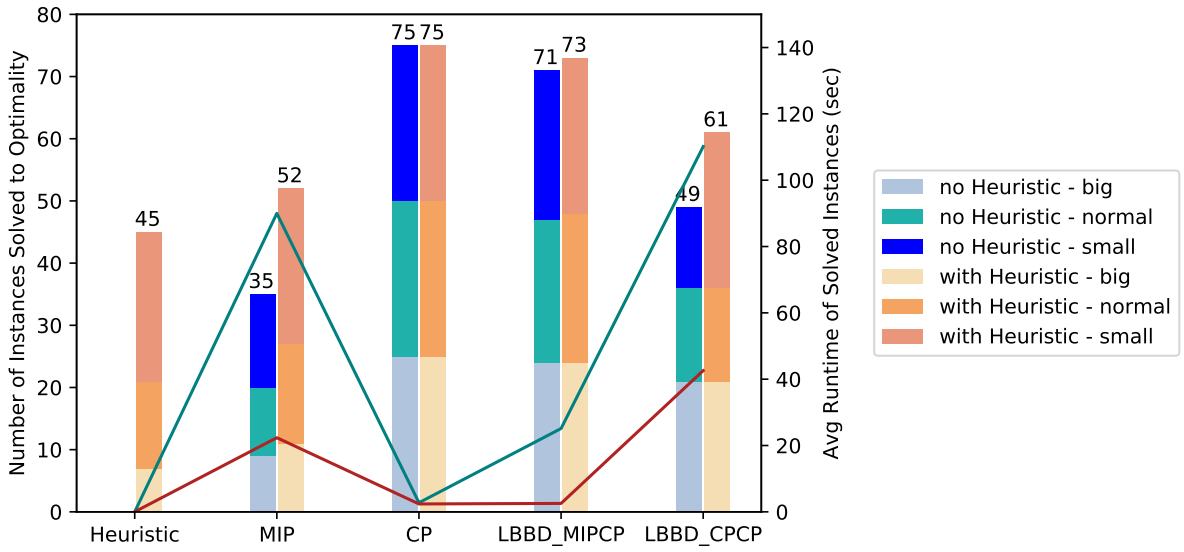


Figure 4.4: Number of instances solved to optimality and average runtime for the generated dataset.

4.5. It can be observed that CP's performance is the best in terms of runtime compared to the three methods. Though MIP/CP LBBD seems to be competitive for some instances that both methods can solve within 0.1 seconds, there are still a large number of instances where MIP/CP LBBD takes more than 10 seconds, or even struggles to solve, but CP takes less than 1 second to solve to optimality. The detailed results are presented in Appendix B.

Based on the above results, we can observe that CP shows exceptionally good performance while the LBBD methods demonstrate a poorer performance. There are two intriguing questions to study: 1) Why do the LLBD approaches work poorly and 2) Why does the CP approach work so well? Empirical analysis suggests that the LBBDs fail in two situations: the instance gets stuck in a subproblem and the algorithm runs out of time in solving that subproblem or the algorithm performs thousands of iterations without finding an optimal solution and thus times out. To answer the second question, we have two orthogonal hypotheses: compared to LBBD approaches, CP finds better quality first solutions faster and the first solutions have a greater impact on search space reduction.

4.3.2 Analysis of LBBD Approaches

As shown in Appendix B, the instances where the LBBD approaches fail have very few iterations for big and normal time windows indicating that the algorithm has encountered a very hard subproblem and times-out trying to solve it. On the other hand, the instances for small time windows show many iterations, which indicates that the master problem is enumerating many possible combinations of request-to-vehicle assignments. The breakdown of runtime in master problem vs subproblem is shown in Figure 4.6 for CP/CP LBBD and MIP/CP LBBD, respectively. For large time window instances, since the vehicle has such a large time window, the master problem assigns all request to a single vehicle, resulting in a subproblem with one vehicle and a lot of requests which in turns results in a very hard subproblem. For small time windows, the percentage of time spent in the master problem is significantly increased as the algorithm goes back and forth between the master problem and the subproblem enumerating many possible master solutions.

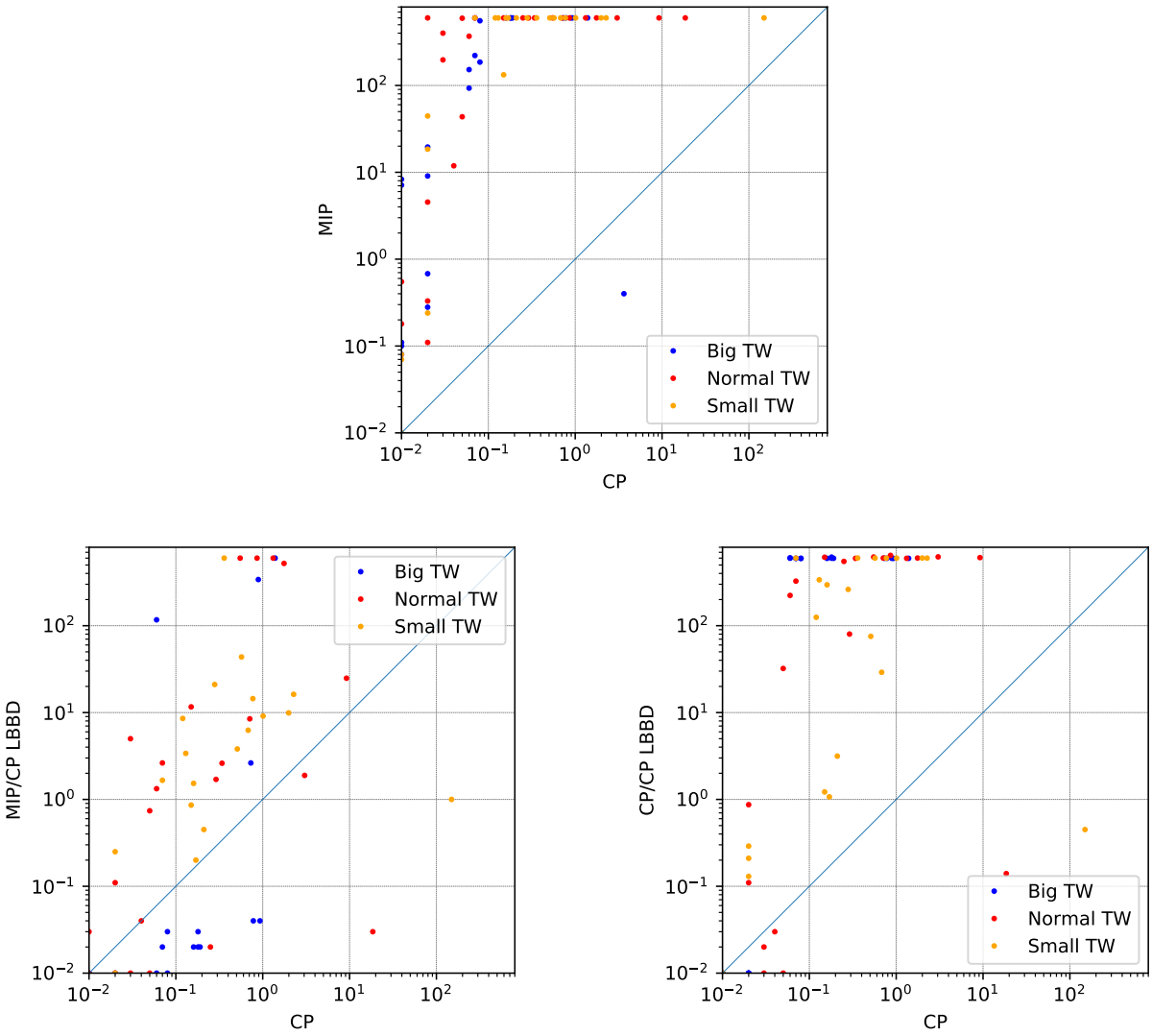
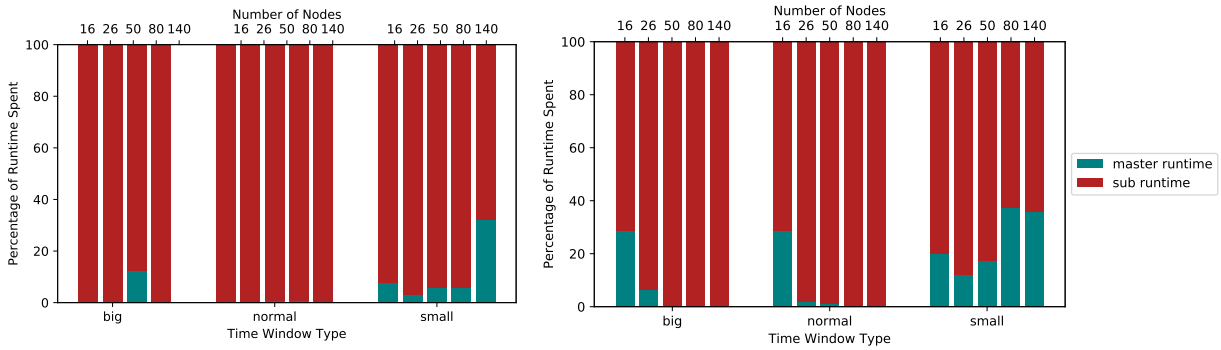


Figure 4.5: Runtime comparison of MIP, CP/CP LBBD, and MIP/CP LBBD to CP.

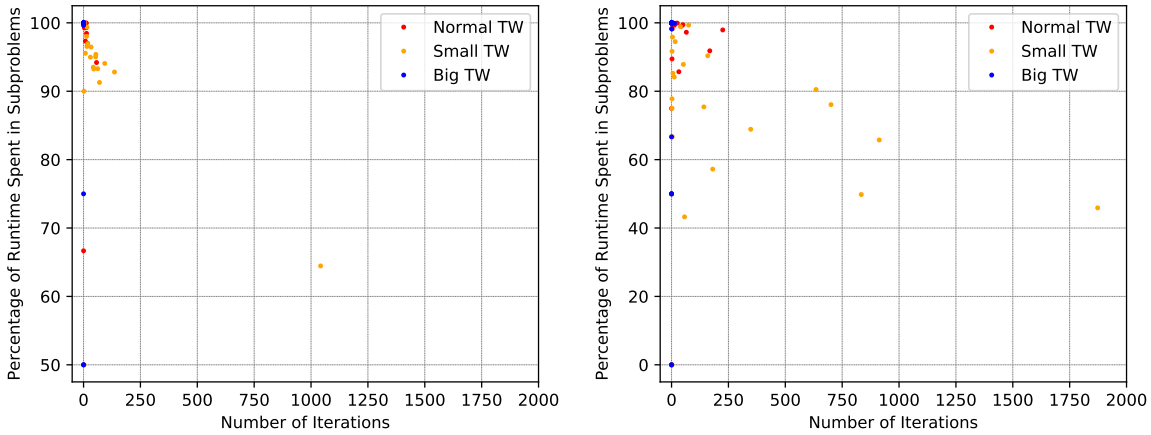


(a) MIP/CP LBBD.

(b) CP/CP LBBD.

Figure 4.6: Comparison of runtime spent in the master problem vs. the subproblem across all instances.

In Figure 4.7, the relationship between the number of iterations and the percentage of runtime spent in the subproblem can be clearly seen. The largest amount of time spent in subproblems occurs when the number of iteration is very small, indicating a very hard subproblem.



(a) MIP/CP LBBD.

(b) CP/CP LBBD.

Figure 4.7: Percentage of time spent in the subproblem vs. the number of iterations.

These results conclude the analysis of the poorer performance of the LBBD approaches. When we decompose the problem, both the master and the subproblems have less information and thus either the master problem makes a poor assignment resulting in a very hard subproblem or there are many iterations between the master problem and the subproblems. More iterations often occur in the small time window instances as the master problem enumerates all possible assignment combinations.

4.3.3 Comparison of the CP Approach and the LBBD Approaches

We have two hypotheses to why the CP model is working so well. First, it seems that given more information to the overall problem, the CP model is able to spread out the distribution of requests to vehicles more "evenly" in initial solutions thus creating better quality solutions and in a shorter amount of time. Further, the solutions that CP finds result in back-propagation from the lower-bound on the objective function, creating a greater impact of search space reduction [56]. In this section, we explore both of these ideas.

CP and Depth First Search

CP Optimizer's default search employs a combination of Large Neighbourhood Search (LNS) and Failure-directed Search (FDS) [67]. To observe its impact, we ran CP on the generated dataset using depth-first search (DFS). All instances were solved to optimality by DFS with an increase in the average runtime from 1.016 seconds to 1.873 seconds, a decrease in the average optimality gap of the first feasible solution from 29.14% to 24.14%, and an increase on the mean time to find the first solution from 0.163 seconds to 0.207 seconds. The difference in using DFS appears marginal, perhaps due to using a single thread in all experiments. However, it does not appear that we can attribute the strong performance of our CP model, relative to the LBBD approaches, to the sophisticated default search of CP Optimizer.

CP and LBBD First Solutions

For this experiment, we run the CP model and both LBBD approaches until finding a first feasible solution. For the CP model, the first solution is the first feasible solution found by the CP Optimizer. The first solution found by the LBBD approaches is the solution found after the first iteration. When the LBBD algorithms encounter a hard subproblem in the first subproblem, then the algorithms return a solution of 0. Otherwise, the algorithms return the routes of scheduled vehicles from previous subproblems as a solution. Two measures are taken from the first solution found: the objective value and the time to find it. The objective value, denoted by z', is compared to the known optimal solution value, z. The first solution gap is computed as $(z - z')/z$. Figures 4.8 and 4.9 show the comparison of the LBBD approaches to the CP model.

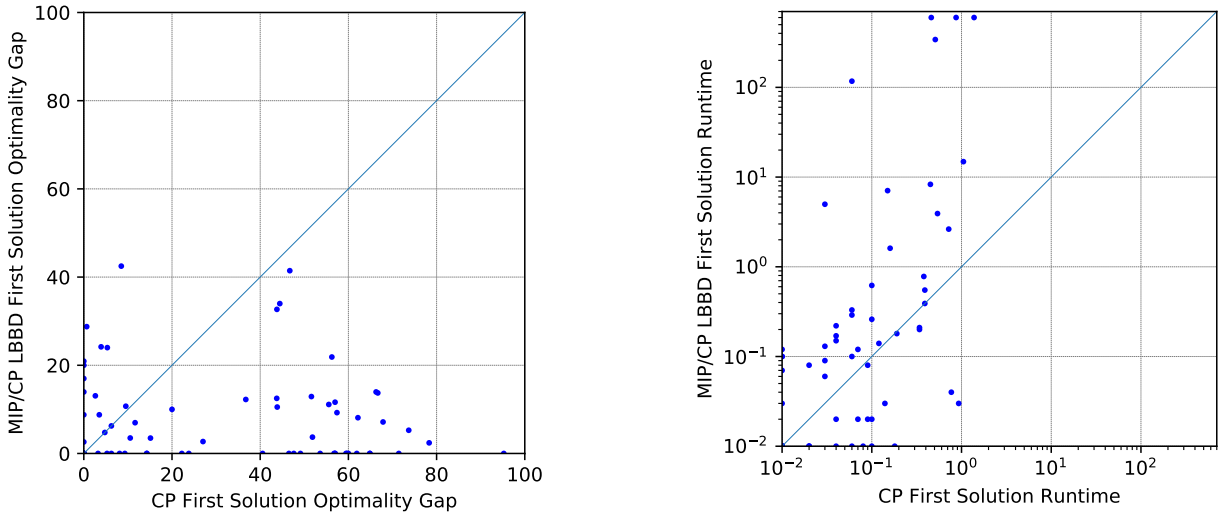


Figure 4.8: First solution quality of MIP/CP LBBD compared to CP.

Due to how the first solution is computed in LBBD approaches, the feasible solution found is often the actual optimal solution. Therefore, while the first solution optimality gap of the LBBD approaches is the same or in many cases smaller than the CP model, the time to find these solutions for the LBBD approaches is larger.

To measure the effect of the first solution on the different approaches, we use the first solution found in CP as a starting solution for the better performing LBBD approach, MIP/CP LBBD. We then let the algorithm run and report the change of runtime with and without the warm start. The warm start solution consists of an assignment of requests to vehicles which is a solution to the master problem of the MIP/CP LBBD, it does not contain any temporal information. The runtime does not include the time to compute the warm start solution. The results are shown in Figure 4.10.

For big time windows, some are solved more quickly with the warm start solution. However, on average, the runtimes with or without the warm-start are the same. In many cases, the warm start solution provided by the CP model is not as good as the first master problem solution and thus the warm start solution is not used.

We conducted the same experiments using the MIP/CP LBBD first solution into the CP model as

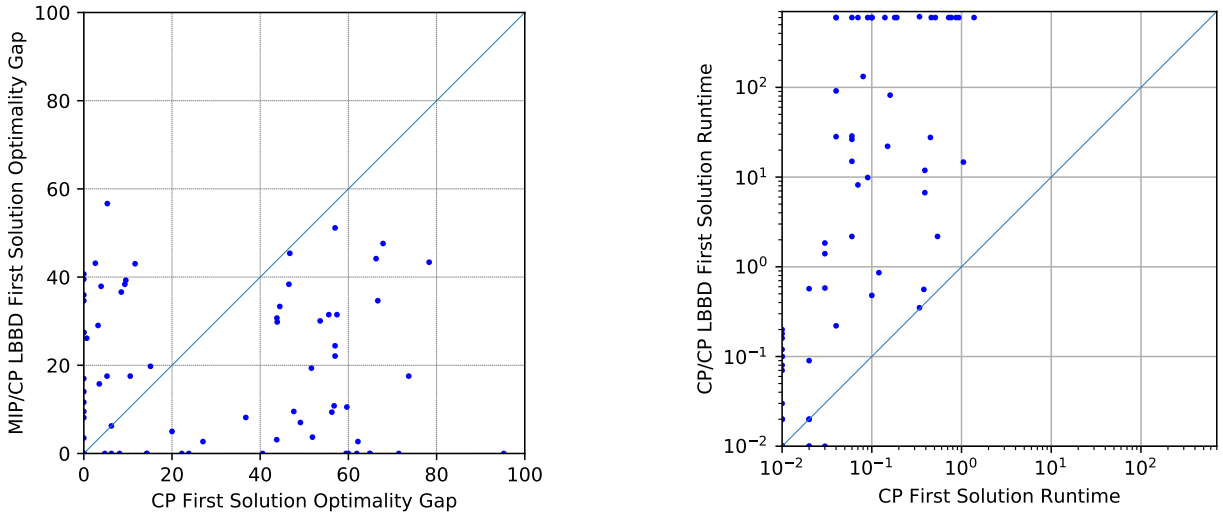


Figure 4.9: First solution quality of CP/CP LBBD compared to CP.

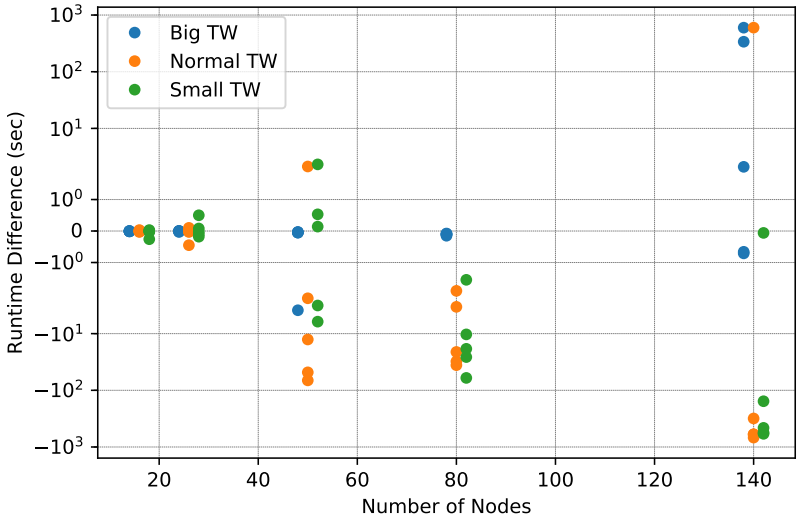


Figure 4.10: Runtime difference of pure MIP/CP LBBD minus MIP/CP LBBD with CP starting solution.

a warm start. The results are shown in Figure 4.11. In this chart, we can clearly see that in most cases, the warm start actually slows down the CP model for the given assignment which supports our hypothesis that the LBBD models find poor first solutions.

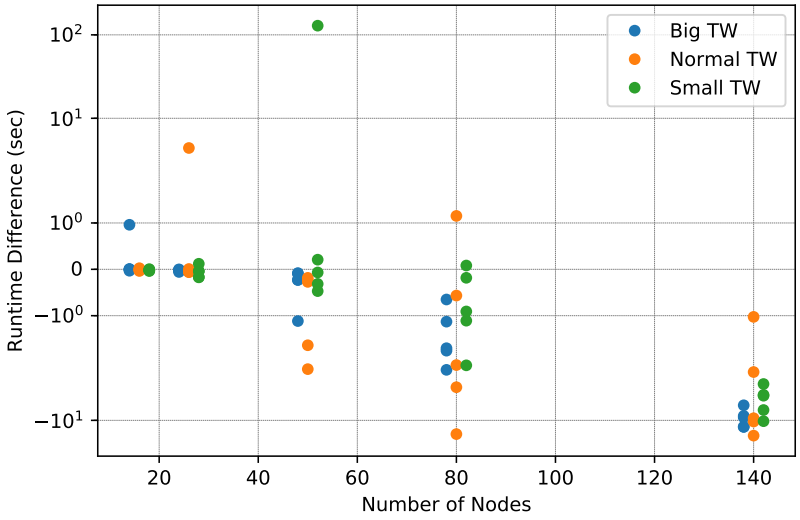


Figure 4.11: Runtime difference of pure CP minus CP with MIP/CP LBBD starting solution.

Search Space Reduction

The next set of experiments measures the impact of search space reduction given artificial lower bounds. If we denote the set of possible values that a variable x_i can take as D_{x_i}, then the logarithm of the size of the search space $\log(|P|)$ is computed as in Equation 4.60.

$$\log(|P|) = \log(|D_{x_i}|) + ... + \log(|D_{x_n}|) \tag{4.60}$$

For interval variables, the domain size is simply the size of the interval (latest finish time - earliest start time) minus the duration of the variable, or $|D_{x_i}| = L_i - E_i - S_i + 1$. The reason for taking the log of the domain sizes is that the domain sizes are often too big. For a fair comparison, we compare CP and CP/CP LBBD to avoid differences caused by different software and methodologies.

Since an optimal solution for each instance is already known, we compute 5 different lower bounds for each dataset that are 100%, 80%, 60%, 40%, and 20% of the actual optimal solution. Note that since we are maximizing, a lower bound on the objective function still results in a feasible solution. We then add this lower bound as a constraint on the objective function for both the CP model and the CP/CP LBBD approach. Both approaches are then allowed to simply propagate without branching, and the search space is calculated. Tables 4.6 and 4.7 show how many instances have shown any search space reduction given the different lower bounds.

There are several instances that have shown a reduction in search space after applying a lower bound indicating that if the CP model finds a good first solution, the search space is also reduced, thus facilitating search. Only one instance demonstrated search space reduction for the CP/CP LBBD

Table 4.6: Number of instances (out of 25 in each row) that have shown a reduction in search space after applying an artificial lower bound for the CP model.

CP		Lower Bound Percentage				
		100%	80%	60%	40%	20%
Time Window Type	small	8	3	2	0	0
	normal	3	1	0	0	0
	big	0	0	0	0	0

Table 4.7: Number of instances (out of 25 in each row) that have shown a reduction in search space after applying an artificial lower bound for the CP/CP LBBD approach.

CP/CP LBBD		Lower Bound Percentage				
		100%	80%	60%	40%	20%
Time Window Type	small	1	1	0	0	0
	normal	0	0	0	0	0
	big	0	0	0	0	0

showing a poor propagation of the first solution quality to the entire search space.

Based on the experimental results, CP is able to find good solutions very quickly as compared to the other methods. Though the search space analysis does not show a significant reduction, we are still confident that the CP algorithm reduces search space more efficiently than the CP/CP LBBD approach. Combining the two major results, CP produces good initial solutions and based on these solutions, CP is able to reduce its search space more resulting in a better performance compared to the LBBD approaches.

4.3.4 CHATS Dataset Results

From the experimental results of the generated dataset, CP and MIP/CP LBBD are the best performing methodologies. The MIP model's size grows too quickly for the large size of CHATS instances and therefore, we only tested CP and MIP/CP LBBD on the CHATS instances. The experiment set-up is the same as for the generated dataset.

Out of the 280 instances, 250 instances solved to optimality with an average of 126.74 seconds using the pure CP model while the MIP/CP LBBD could only solve 47 instances with an average of 331.31 seconds. All other instances timed out without finding the optimal solution or proving optimality. Figure 4.12 outlines the summary results of the CHATS dataset.

Taking a closer look at the CP runtime shown in Figure 4.13, it can clearly be seen that as the instances get larger, the runtime is longer.

Since CP shows the best performance, we also let the CP algorithm run for 8 hours in an attempt to find an actual optimal solution of the unsolved instances. 21 instances were solved to optimality but 9 instances are still unsolved. Thus the best known solution is reported and the gap is calculated. The overall mean optimality gap for CP is 5.25% and 11.84% for MIP/CP LBBD. For the 233 instances that are unsolved by MIP/CP LBBD, the average optimality gap is 14.23% for MIP/CP LBBD and 6.31% for pure CP. Furthermore, for the 30 instances that CP failed to find optimal solutions in the 600 seconds time limit, MIP/CP LBBD finds better solutions and the average gap to the best known solution is

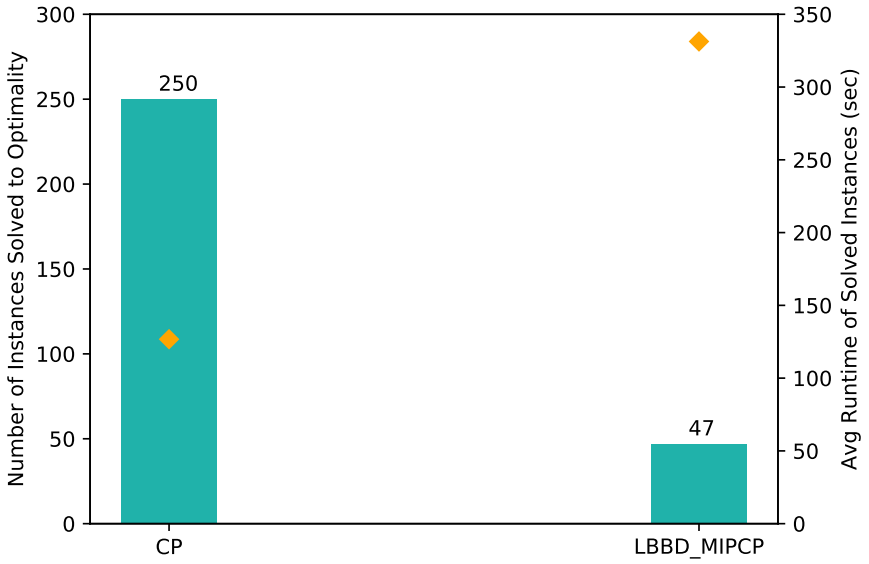


Figure 4.12: Number of instances solved to optimality and average runtime for the CHATS dataset.

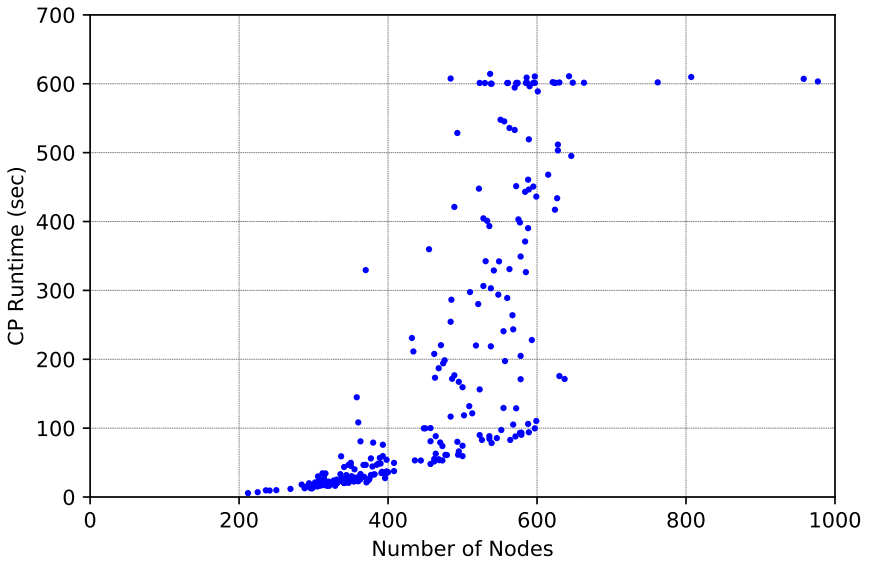


Figure 4.13: CP runtime of CHATS instances over number of vertices.

49.02% for CP while the gap is only 18.38% for MIP/CP LBBD. Out of these 30 instances, MIP/CP finds better solutions than CP in 22 instances. The optimality gap information is outlined in Table 4.8.

Table 4.8: Average optimality gap summary for CP and MIP/CP LBBD on CHATS instances.

Instances	CP avg gap	MIP/CP LBBD avg gap
All 280 instances	**5.25%**	11.84%
233 instances not solved by MIP/CP LBBD	**6.31%**	14.23%
30 instances not solved by CP	49.02%	**18.38%**

Overall, CP finds and proves optimal solutions or it does not find a good solution, while MIP/CP LBBD finds fairly good solutions in most instances. The optimality gap comparison is shown in Figure 4.14.

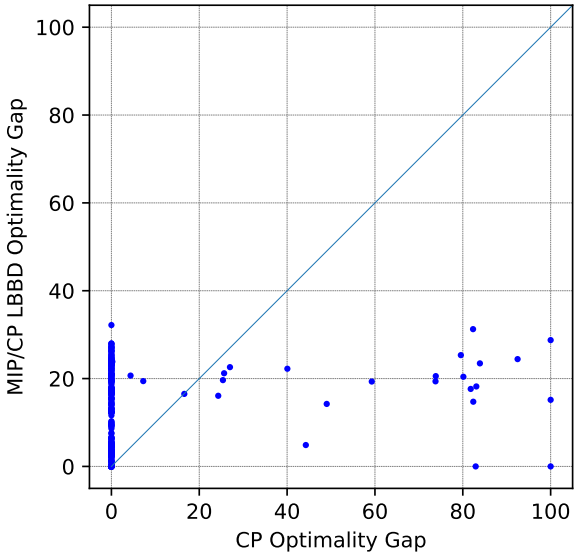


Figure 4.14: Optimality gap comparison between CP and MIP/CP LBBD on CHATS instances.

4.4 Conclusions

Inspired from real-world problems, the Senior Transportation Problem (STP) is a combination of Pickup and Delivery Problem with Time Windows, Dial-a-Ride Problem, and Team Orienteering Problem and proves to be a hard problem to solve. In this chapter, a formal problem definition for the STP has been proposed illustrating multiple constraints inspired from real-life problems.

Five different approaches, Mixed Integer Programming (MIP), Constraint Programming (CP), MIP/CP Logic-based Benders Decomposition (LBBD), CP/CP LBBD, and a construction heuristic, solving the STP have been developed. The variables used, the mathematical formulations and pseudo-code are presented. Each method is tested on 75 instances from the generated dataset as described in the previous chapter. CP proves to be the best performing algorithm both in terms of the number of instances solved to proven optimality and faster runtime, followed by MIP/CP LBBD, CP/CP LBBD, construction

heuristic, then MIP. In order to explain the good performance of CP compared to the LBBD approaches, experiments were performed to investigate two hypotheses: compared to LBBD approaches, CP finds better first solutions and the solutions have a greater impact on search space reduction. Furthermore, analysis on the LBBD methods shows that they fail in two situations: the algorithm gets stuck in a hard subproblem due to a poor assignment from the master problem, or the algorithm goes through too many iterations.

The two best performing algorithms, CP and MIP/CP LBBD are then tested on CHATS instances. Again, CP proved to be the best algorithm, solving 250 out 280 instances to optimality while MIP/CP LBBD only solved 47 instances to optimality in the given 600 seconds time limit.

Chapter 5

The STP with Overbooking

Our partner organization, CHATS, indicated that one of the major problems that they face is cancellations. They experience roughly 20% cancellations on their scheduled requests, which leads us to explore an extension of the Senior Transportation Problem (STP) and the possibility of overbooking requests. In this chapter, we provide the definition of the Senior Transportation Problem with Overbooking (STPOB) and provide five solution techniques in a similar structure as the previous chapter. The different techniques are then evaluated using both the generated dataset and CHATS dataset as described in Chapter 3. Detailed analysis on the different methods are performed in order to demonstrate the reasons why one methodology performs better than another.

5.1 Problem Definition

As described in Chapter 2, the notion of overbooking has not been previously studied in transportation problems. Therefore, we borrow some ideas from the clinical appointment scheduling overbooking problem [50] where the authors computed cancellation probabilities for each appointment and created a schedule with overlapping appointments. The objective is to maximize the expected profits gained from appointments that actually occur while minimizing the wait time and the clinic overtime. For the STPOB, we also compute probabilities of cancellation and use this information as input data. However, rather than booking overlapping appointments, we introduce a new unconstrained vehicle that can serve overbooked requests at a higher cost. The objective is simply minimizing the total expected cost.

In the previous chapter, we maximized the number of requests that could be served given the available vehicles. Whereas, in the STPOB, we assume that all requests must be served, each with an additional information of probability of cancellation. To compensate for the requests that cannot be served by the given vehicles, we introduce one "taxi" vehicle that can hypothetically fulfil all requests simultaneously. Additionally, each vehicle is associated with a vehicle cost factor, where the cost of volunteer vehicles is less than the cost of paid drivers, which are both substantially less than the cost of the taxi vehicles. The objective of the STPOB is then to minimize the total expected cost of scheduling all requests. The STPOB can be defined as an extension of the STP: given the same set of constraints as the STP (see Section 4.1), the STPOB minimizes the total expected cost of serving all requests with the addition of higher cost taxi vehicles which are completely unconstrained. The expectation arises from each trip having a probability of cancellation and the assumption that cancelled trips do not incur any cost.

The STPOB differs from the STP in the objective function, the addition of taxi vehicles, a cancellation probability π_r associated with each request r, and a cost factor $costFactor_k$ attribute associated with each vehicle k. The STPOB is still described with the same graph G (see Section 4.1) with the addition of a taxi node k^* which represents an infinite number of taxi vehicles. The structure of the rest of the request and vehicle information remains the same as the STP.

When a request cannot be completed by a volunteer driver or a paid driver, a higher cost taxi vehicle will complete the request. It is assumed that for the taxi vehicle, there are no constraints; it can satisfy any request. We then redefine the set of vehicles $\mathcal{K}^*$ as $\mathcal{K} \cup k^*$ where k^* is the taxi vehicle. The taxi vehicle has no restriction on earliest start time or latest finish time, has unlimited capacity and can start and end at any location.

Each request r is associated with a total travel time B_r, and each vehicle is associated with a vehicle cost factor V_k such that $V_{taxi} > V_{paiddriver} > V_{volunteer}$. The cost that is associated with each request is then its distance multiplied by the vehicle cost factor and by the probability of the request not being cancelled. The mathematical representation is shown in Equation (5.1). For example, if request r is assigned to vehicle k, then the cost of request r, O_r is as below.

$$O_r = (1 - \pi_r) \times B_r \times V_k \tag{5.1}$$

The cost O_r is only an estimate; the true of cost of a trip also includes gas prices which are dependent on the total distance travelled. We have converted all cost factors to be based on the travel time in order to facilitate computation. Examinations of true costs can be studied further in future work. The new parameters are summarized in Table 5.1. A list of all parameters can be found in Appendix A.

Table 5.1: Additional parameters for the STPOB.

Parameter	Description
π_r	Probability of cancellation of request r
B_r	Total travel time length of request r. $B_r = T_{i_{r^+}, i_{r^-}}$
V_k	Cost factor of vehicle k. $V_{taxi} > V_{paiddriver} > V_{volunteer}$
O_i	Expected cost of request r

5.2 Solution Approaches for the STPOB

In this section, we present five different approaches to solve the STPOB. As in the previous chapter, we present a MIP approach, a CP approach, followed by MIP/CP and CP/CP LBBD approaches and we complete this section by describing a simple construction heuristic.

5.2.1 A MIP Approach

The variables for the MIP model are the same as in the MIP model for the STP (see Section 4.2.1) with the addition of the variables associated with the taxi vehicle. Most of the constraints are the same except for the addition of an unconstrained taxi vehicle and the change of the objective function. A

MIP formulation is presented below.

$$\min \sum_{k\in\mathcal{K}^*}\sum_{r\in\mathcal{R}}\sum_{j\in\mathcal{V}}\left(x_{k,i_{r^+},j}\times(1-\pi_r)\times B_r\times V_k\right) \tag{5.2}$$

$$\text{s.t.} \sum_{j\in\mathcal{N}^+} x_{k,i_{k^+},j}+x_{k,i_{k^+},i_{k^-}}=1 \qquad \forall k\in\mathcal{K} \tag{5.3}$$

$$\sum_{i\in\mathcal{N}^-} x_{k,i,j_{k^-}}+x_{k,i_{k^+},i_{k^-}}=1 \qquad \forall k\in\mathcal{K} \tag{5.4}$$

$$\sum_{k\in\mathcal{K}^*}\sum_{j\in\mathcal{V}} x_{k,i_{r^+},j}=1 \qquad \forall r\in\mathcal{R} \tag{5.5}$$

$$\sum_{j\in\mathcal{V}}(x_{k,i,j}-x_{k,j,i})=0 \qquad \forall k\in\mathcal{K}^*, i\in\mathcal{N} \tag{5.6}$$

$$\sum_{j\in\mathcal{V}}\left(x_{k,i_{r^+},j}-x_{k,j,i_{r^-}}\right)=0 \qquad \forall k\in\mathcal{K}^*, r\in\mathcal{R} \tag{5.7}$$

$$u_{k,j}\geq(u_{k,i}+T_{i,j}+S_j)-M\times(1-x_{k,i,j}) \qquad \forall k\in\mathcal{K}, i,j\in\mathcal{V} \tag{5.8}$$

$$u_{k,i}\geq E_i-M\times\left(1-\sum_{j\in\mathcal{V}}x_{k,i,j}\right) \qquad \forall k\in\mathcal{K}, i\in\mathcal{V} \tag{5.9}$$

$$u_{k,i}\leq L_i-S_i+M\times\left(1-\sum_{j\in\mathcal{V}}x_{k,i,j}\right) \qquad \forall k\in\mathcal{K}, i\in\mathcal{V} \tag{5.10}$$

$$u_{k,i_{r^+}}\leq u_{k,i_{r^-}} \qquad \forall k\in\mathcal{K}, r\in\mathcal{R} \tag{5.11}$$

$$\left(u_{k,i_{r^-}}-u_{k,i_{r^+}}\right)\leq F \qquad \forall k\in\mathcal{K}, r\in\mathcal{R} \tag{5.12}$$

$$v_{k,j}\geq(v_{k,i}+Q_i)-M\times(1-x_{k,i,j}) \qquad \forall k\in\mathcal{K}, i,j\in\mathcal{V} \tag{5.13}$$

$$x_{k,i,j}\in\{0,1\} \qquad \forall k\in\mathcal{K}, (i,j)\in\mathcal{A} \tag{5.14}$$

$$0\leq u_{k,i}\leq Z \qquad \forall k\in\mathcal{K}, i\in\mathcal{V} \tag{5.15}$$

$$0\leq v_{k,i}\leq C_k \qquad \forall k\in\mathcal{K}, i\in\mathcal{V} \tag{5.16}$$

The objective function (5.2) has been changed to minimize the total expected cost of satisfying all requests. For each request r that is assigned to vehicle k, the expected cost associated with request i is the probability of the request happening multiplied by the cost factor of the vehicle k and by the total travel length of the request r.

Constraints (5.3) and (5.4) ensure that each vehicle leaves from its depot location to either a request's pickup place or to its associated ending depot directly. Constraints (5.5) is changed from an inequality to an equality as each request must be satisfied by a vehicle including the taxi vehicle. The constant flow is again ensured through Constraints (5.6), with the taxi vehicle added. Constraints (5.7) ensures that the pickup and delivery locations of the same request are visited by the same vehicle including the taxi vehicle.

The time window, capacity, variable bound constraints (5.8) - (5.16) are the same as for the STP as the taxi vehicle is not subject to these constraints.

5.2.2 A CP Approach

Again, we use the same set of variables as from the CP model of the STP (see Section 4.2.2) with additional variables for the taxi vehicle. All location variables x_i are set to *non*-optional since all locations will be visited by a vehicle. Below is a CP approach to solve the STPOB.

$$\text{minimize} \sum_{k \in \mathcal{K}^*} \sum_{r \in \mathcal{R}} \left(\texttt{PresenceOf}(\bar{X}_{k,i_{r^+}}) \times (1 - \pi_r) \times V_k \times B_r \right) \tag{5.17}$$

subject to

Location Constraints

$$\texttt{Alternative}(x_i, \mathbf{X_i}) \qquad \forall i \in \mathcal{N} \tag{5.18}$$

$$\texttt{Before}(\bar{X}_{k,i_{r^+}}, \bar{X}_{k,i_{r^+}}) \qquad \forall k \in \mathcal{K}, \forall r \in \mathcal{R} \tag{5.19}$$

$$\texttt{PresenceOf}(\bar{X}_{k,i_{r^+}}) = \texttt{PresenceOf}(\bar{X}_{k,i_{r^-}}) \qquad \forall k \in \mathcal{K}^*, \forall r \in \mathcal{R} \tag{5.20}$$

Ride Time Constraint

$$\texttt{GetStartMax}(x_{i_{r^-}}) - \texttt{GetEndMax}(x_{i_{r^+}}) \leq F \qquad \forall r \in \mathcal{R} \tag{5.21}$$

Capacity Constraints

$$v_{k,i} = \texttt{StepAtStart}(\bar{X}_{k,i}, Q_i) \qquad \forall k \in \mathcal{K}, \forall i \in \mathcal{N} \tag{5.22}$$

$$\sum_{i \in \mathcal{N}} v_{k,i} \leq C_k \qquad \forall k \in \mathcal{K} \tag{5.23}$$

Route Constraints

$$\texttt{First}(u_k, x_{i_{k^+}}) \qquad \forall k \in \mathcal{K} \tag{5.24}$$

$$\texttt{Last}(u_k, x_{i_{k^-}}) \qquad \forall k \in \mathcal{K} \tag{5.25}$$

$$\texttt{NoOverlap}(u_k, tt) \qquad \forall k \in \mathcal{K} \tag{5.26}$$

Although all requests must be satisfied, depending on which vehicle the request is assigned to, the associated expected cost varies. Therefore, in the objective function (5.17), the variables bounded by the `PresenceOf` constraint are changed to the assignment variables $\bar{X}_{k,i_{r^+}}$ from the location variable x_i since all locations will be visited by a vehicle. If vehicle k serves request r, the expected cost of the request is the total travel length of the request multiplied by the vehicle cost factor and the probability that the request will happen.

Constraints (5.18) to (5.26) are the same as in the CP model for the STP (see Section 4.2.2) except for Constraints (5.20) which is modified for the taxi vehicle. The constraints restrict pickup and delivery locations of a request to be visited by the same vehicle (including the taxi vehicle).

5.2.3 Logic-based Benders Approaches

An LBBD approach has also been developed to solve the STPOB. The idea remains the same as in Chapter 4: the master problem assigns requests to vehicles including the taxi. Each vehicle then solves a minimization problem to find its minimal cost route. If the objective cost returned by a subproblem is higher than the cost assigned by the master problem, a Benders cut is returned to the master problem. For the next iteration, the master problem has a newly added constraint that states that the cost of

assigning these specific requests onto this vehicle has to be at least what the subproblem has returned in the previous iteration.

MIP Master Problem

The MIP master problem employs the same variables as for the MIP master for the STP (see Section 4.2.3) with additional variables for the taxi vehicle.

$$\text{minimize} \sum_{k \in \mathcal{K}^*} \sum_{r \in \mathcal{R}} (\varphi_{k,r} \times (1 - \pi_r) \times B_r \times V_k) \tag{5.27}$$

$$\text{subject to} \sum_{k \in \mathcal{K}^*} y_{k,i} = 1 \qquad \forall i \in \mathcal{N} \tag{5.28}$$

$$Q_r \times \varphi_{k,r} \leq P_k \qquad \forall k \in \mathcal{K}, r \in \mathcal{R} \tag{5.29}$$

$$\zeta_r = S_{i_{r^+}} + T_{i_{r^+}, i_{r^-}} + S_{i_{r^-}} \qquad \forall r \in \mathcal{R} \tag{5.30}$$

$$(E_{i_{r^+}} + \zeta_r) \times \varphi_{k,r} \leq L_{i_{k^-}} \qquad \forall k \in \mathcal{K}, r \in \mathcal{R} \tag{5.31}$$

$$L_{i_{r^-}} - \zeta_r \geq E_{i_{k^+}} \times \varphi_{k,r} \qquad \forall k \in \mathcal{K}, r \in \mathcal{R} \tag{5.32}$$

$$\sum_{i \in \mathcal{N}} (y_{k,i} \times \mathcal{T}_i + S_i) + \mathcal{T}_{i_{k^+}} + S_{i_{k^+}} \leq L_{i_{k^-}} - E_{i_{k^+}} \qquad \forall k \in \mathcal{K} \tag{5.33}$$

$$y_{k,r} = y_{k,r+|\mathcal{R}|} = \varphi_{k,r} \qquad \forall k \in \mathcal{K}^*, r \in \mathcal{R} \tag{5.34}$$

$$y_{k,i}, \varphi_{k,r} \in \{0, 1\} \qquad \forall k \in \mathcal{K}^*, i \in \mathcal{N}, r \in \mathcal{R} \tag{5.35}$$

Benders Cuts

The objective function (5.27) has been changed to minimize the total expected costs. Constraints (5.28), (5.34), and (5.35) have been altered to include the taxi vehicle, while the rest of the constraints remain the same as for the STP.

CP Master Problem

All variables remain the same as the CP master problem for the STP (see Section 4.2.3) with the addition of variables associated with the taxi vehicle. All location variables x_i are set to *non*-optional. The CP master problem for the STPOB is presented below.

$$\text{minimize} \sum_{k \in \mathcal{K}^*} \sum_{r \in \mathcal{R}} \left(\texttt{PresenceOf}(X_{i_{r^+},k}) \times (1 - \pi_r) \times V_k \times B_r \right) \tag{5.36}$$

subject to

Location Constraints

$$\texttt{Alternative}(x_i, \mathbf{X}_i) \qquad \forall i \in \mathcal{N} \tag{5.37}$$

$$\texttt{PresenceOf}(X_{i_{r^+},k}) = \texttt{PresenceOf}(X_{i_{r^-},k}) \qquad \forall k \in \mathcal{K}^*, r \in \mathcal{R} \tag{5.38}$$

Route Constraints

$$\texttt{EndBeforeStart}(x_{i_{k^+}}, X_{i,k}) \qquad \forall k \in \mathcal{K}, i \in \mathcal{N} \tag{5.39}$$

$$\texttt{EndBeforeStart}(X_{i,k}, x_{i_{k^-}}) \qquad \forall k \in \mathcal{K}, i \in \mathcal{N} \tag{5.40}$$

$$\texttt{EndBeforeStart}(x_{i_{k^+}}, x_{i_{k^-}}) \qquad \forall k \in \mathcal{K} \tag{5.41}$$

Distance Constraint

$$\sum_{i \in \mathcal{N}} (\texttt{PresenceOf}(X_{i,k}) \times \mathcal{T}_i + S_i) + \mathcal{T}_{i_{k^+}} + S_{i_{k^+}} \leq L_{i_{k^-}} - E_{i_{k^+}} \qquad \forall k \in \mathcal{K} \tag{5.42}$$

Benders Cuts

Similar to the previous models, the objective function has been changed and the taxi vehicle has been added to Constraints (5.38) while keeping all other constraints the same.

CP Subproblem

The CP subproblem employs the same variables as the CP subproblem for the STP (see Section 4.2.3). Since a subproblem may not be able to serve all requests that the master problem has assigned to it and since we would like to compute an upper bound on the cost of all the requests assigned to a specific vehicle, a taxi vehicle must also be added to each subproblem. Each subproblem then forms a scheduling optimization problem for a two-vehicle problem, thus we will be in need of $X_{i,k}$ variables, as in the master problem, to find the assignment of requests to either the organization's vehicle or to the taxi vehicle. The new variables are only used to compute the total cost and are only constrained to restrict pickup and delivery locations to be visited by the same vehicle. The variables employed in the subproblem are outlined in Table 5.2.

A CP model for the subproblem of the LBBD approaches for the STPOB is given in (5.43) - (5.52).

Table 5.2: Variables for vehicle-based model.

Location Variables	**Type**	**Description**
x_i	interval	Time interval in which location $i \in \mathcal{V}^*$ will be served. The variable is bounded by the time window at location i.
$X_{i,k}$	interval	Time interval in which location $i \in \mathcal{N}$ is visited by vehicle k and not present if location i is not visited by vehicle k. $k = 0$ if vehicle associated with the subproblem, and $k = 1$ if taxi vehicle.
Load Variables	**Type**	**Description**
v_i	cumul function	Load of vehicle $k = 0$ after visiting node $i \in \mathcal{N}^*$
Sequence Variable	**Type**	**Description**
u	sequence	Sequence of locations visited by vehicle $k = 0$

$$\text{minimize} \sum_{k \in \{0,1\}} \sum_{r \in \mathcal{R}^*} \left(\texttt{PresenceOf}(X_{i_{r^+},k}) \times (1 - \pi_r) \times V_k \times B_r \right) \tag{5.43}$$

suject to

Location Constraints

$$\texttt{Alternative}(x_i, X_i) \qquad \forall i \in \mathcal{N}^* \tag{5.44}$$

$$\texttt{Before}(X_{i_{r^+},k}, X_{i_{r^-},k}) \qquad \forall k \in \mathcal{K}, \forall r \in \mathcal{R}^* \tag{5.45}$$

$$\texttt{PresenceOf}(X_{i_{r^+},k}) = \texttt{PresenceOf}(X_{i_{r^-},k}) \qquad \forall k \in \mathcal{K}^*, \forall r \in \mathcal{R}^* \tag{5.46}$$

Ride Time Constraints

$$\texttt{GetStartMax}(x_{i_{r^-}}) - \texttt{GetEndMax}(x_{i_{r^+}}) \leq F \qquad \forall r \in \mathcal{R}^* \tag{5.47}$$

Capacity Constraints

$$v_i = \texttt{StepAtStart}(v_i, Q_i) \qquad \forall i \in \mathcal{N}^* \tag{5.48}$$

$$0 \leq \sum_{i \in \mathcal{N}^*} v_i \leq C_{k^*} \tag{5.49}$$

Route Constraints

$$\texttt{First}(u, x_{i_{k^*+}}) \tag{5.50}$$

$$\texttt{Last}(u, x_{i_{k^*-}}) \tag{5.51}$$

$$\texttt{NoOverlap}(u) \tag{5.52}$$

The subproblem finds the minimal cost schedule given the set of requests and the specific vehicle with the addition of a taxi vehicle. The objective function (5.43) has been altered to reflect the minimal cost while the rest of the constraints are left unchanged. All other constraints are similar to that of the global CP model.

Benders Cut

Each subproblem minimizes the total cost of scheduling the given requests to the two vehicles, the organization vehicle and a taxi. Since the taxi is always more costly than the organization vehicle, when the minimal cost of the subproblem does not equal to the cost computed by the master problem, the schedule had to employ the more expensive taxi. The cost incurred by the organization's vehicle then becomes an upper bound on the cost of the given requests when assigned to the organization's vehicle. The upper bound on the cost of the given requests are returned to the master problem as the Benders cut, forcing the master problem to assign a subset of requests whose total cost is smaller or equal to that upper bound or a different set of requests. In this subsection, we present two lemmas, one showing that the cost of the organization vehicle in each subproblem represents an upper bound on the given requests assigned to the vehicle and another showing that the cut does not remove feasible solutions. Finally, the two lemmas result in a theorem showing the validity of the Benders cut presented.

First, we will present some notation. In each subproblem, let k_0 represent the organization's vehicle, and let k_1 represent the taxi vehicle. Then, $c = \sum_{k \in \{k_0, k_1\}} \sum_{r \in \mathcal{R}^*} \left(\texttt{PresenceOf}(X_{i_{r^+},k}) \times (1 - \pi_r) \times V_k \times B_r \right)$ is the minimum cost of assigning the requests $r \in \mathcal{R}^*$ to k_0 and k_1 found by the subproblem. Let $\mathcal{R}_0^*$ represent the set of requests assigned to k_0 or and $\mathcal{R}_1^*$ represent the set of requests assigned to k_1 in the optimal schedule of the subproblem. Accordingly, $z'_k = \sum_{r \in \mathcal{R}_k^*} ((1 - \pi_r) \times B_r) \times V_k$ is the cost of all requests assigned to vehicle k in the subproblem. Thus, c can be represented as $z'_0 + z'_1$. We will show that z'_0 is an upper bound of the requests assigned to k_0 through Lemma 1.

Lemma 1. *Given a set of request $\mathcal{R}^*$ and the minimum cost, $c = z'_0 + z'_1$, of assigning these requests to a vehicle k_0 and a taxi vehicle k_1, then z'_0 is an upper bound to the minimum cost of assigning requests to k_0.*

Proof. By contradiction.

In this proof, we consider three cases that may induce changes to z'_0: 1. move a request $r_i \in \mathcal{R}_1^*$ from the k_1 to k_0, 2. move a request $r_j \in \mathcal{R}_0^*$ from k_0 to k_1 and 3. swap requests $r_i \in \mathcal{R}_1^*$ and $r_j \in \mathcal{R}_0^*$ and show that none of the actions can produce a cost $\bar{z}'_0 > z'_0$. Let $\bar{c} = \bar{z}'_0 + \bar{z}'_1$ be cost of reassigning requests and $c = z'_0 + z'_1$ be cost of the original schedule and minimal cost of assigning requests $\mathcal{R}^*$. Consider any request r, its cost is calculated as $(1 - \pi_r) \times B_r \times V_k$. For ease of notation, we denote $(1 - \pi_r) \times B_r$ as δ_r. We are given that $V_1 > V_0$.

1. Suppose there exists $r_i \in \mathcal{R}_1^*$ such that by assigning it to k_0 instead of k_1, $\bar{z}'_0 > z'_0$. If r_i if moved to k_0, the cost of the subproblem will be reduced. However, because c is the minimum cost, there cannot exist another schedule with cost smaller than c, thus we have derived a contradiction. Such r_i does not exist and z'_0 is indeed the maximum cost of requests assigned to k_0.

2. Moving a request $r_j \in \mathcal{R}_0^*$ from k_0 to k_1 would reduce the cost of k_0, however would increase the cost of k_1 by a strictly larger amount since the cost of the k_1 is strictly larger than k_0. This will result in a larger c.

3. Suppose there exist requests $r_i \in \mathcal{R}_1^*$ and $r_j \in \mathcal{R}_0^*$ such that by swapping their assigned vehicles,

$\bar{z}'_0 > z'_0$. Then,

$$\begin{aligned}
\bar{z}'_0 &= z'_0 + \delta_{r_i} \times V_0 - \delta_{r_j} \times V_0 \\
&= z'_0 + (\delta_{r_i} - \delta_{r_j}) \times V_0 \\
\because \bar{z}'_0 > z'_0 \text{ and } V_0 > 0 \therefore (\delta_{r_i} - \delta_{r_j}) &> 0
\end{aligned}$$

$$\begin{aligned}
\bar{z}'_1 &= z'_1 - \delta_{r_i} \times V_1 + \delta_{r_j} \times V_1 \\
&= z'_1 - (\delta_{r_i} - \delta_{r_j}) \times V_1 \\
\bar{c} &= z'_0 + z'_1 + (\delta_{r_i} - \delta_{r_j}) \times (V_0 - V_1) \\
&= c + (\delta_{r_i} - \delta_{r_j}) \times (V_0 - V_1) \\
&< c \text{ (since } V_1 > V_0 \text{ and } (\delta_{r_i} - \delta_{r_j}) > 0).
\end{aligned}$$

Since c is the minimal cost, there cannot exist such a pair, thus we have derived a contradiction. Such r_i and r_j do not exist and z'_0 is the maximum cost.

This concludes the proof that z'_0 is the maximum cost of assigning requests R^* to k_0. □

Given the upper bound z'_0, the Benders cut returned to the master problem is modelled in the MIP formulation as Equation (5.53) and in the CP formulation as Equation (5.54).

$$\sum_{r \in \mathcal{J}_{h,k}} (\varphi_{k,r} \times (1 - \pi_r) \times B_r \times V_k) \leq z'_0 \qquad \forall k \in \mathcal{K}, h \in \{1, ..., H-1\} \tag{5.53}$$

$$\sum_{r \in \mathcal{J}_{h,k}} (\texttt{PresenceOf}(X_{i_{r^+},k}) \times (1 - \pi_r) \times B_r \times V_k) \leq z'_0 \qquad \forall k \in \mathcal{K}, h \in \{1, ..., H-1\} \tag{5.54}$$

Next, we show that the Benders cuts presented above do not remove any global feasible solutions.

Lemma 2. *The proposed cuts shown in Equations* (5.53) *and* (5.54) *do not eliminate any global feasible solutions.*

Proof. When a subproblem must employ a taxi, it is not feasible to schedule all requests assigned to it on the vehicle and for the purpose of this proof, we will define this as an infeasible schedule. Let $\mathcal{R}_k^{(h)}$ be the set of requests assigned to vehicle k by the master problem at iteration h with cost w_k that is less than the optimal cost $c_{\mathcal{R}_k^{(h)},k}$ of the subproblem associated with vehicle k, $w_k < c_{\mathcal{R}_k^{(h)},k}$. Then the returned Benders cut will be $\sum_{r \in \mathcal{R}_k^{(h)}} (\varphi_{k,r} \times (1 - \pi_r) \times B_r \times V_k) \leq z'_0$. Similar to Lemma 1, we denote $(1 - \pi_r) \times B_r$ as δ_r. Consider a set of requests $\mathcal{R}_k^*$ with an optimal cost of $c_{\mathcal{R}_k^*,k}$, we show that $\mathcal{R}_k^*$ satisfies the inequality in Equations (5.53) and (5.54) if it is feasible. In this proof, we employ a similar approach as Roshanaei [58], we present five possible cases of $\mathcal{R}_k^*$.

1. In $\mathcal{R}_k^*$, some more requests $\bar{\mathcal{R}}_k^*$ have been added to $\mathcal{R}_k^{(h)}$.

$$\begin{aligned} c_{\mathcal{R}_k^*,k} &= \sum_{r \in \mathcal{R}_k^{(h)} \cup \bar{\mathcal{R}}_k^*} \varphi_{k,r} \times \delta_r \times V_0 \\ &= \sum_{r \in \mathcal{R}_k^{(h)}} \varphi_{k,r} \times \delta_r \times V_0 + \sum_{r \in \bar{\mathcal{R}}_k^*} \varphi_{k,r} \times \delta_r \times V_0 \\ &> z_0' \end{aligned}$$

   Since $\mathcal{R}_k^{(h)}$ is already incurring more cost than w_k, a taxi must have been employed, adding more requests will still need to employ a taxi, thus $R_k^{(h)}$ does not have a feasible solution and as shown above does not satisfy the Benders cut.

2. In $\mathcal{R}_k^*$, some requests $\bar{\mathcal{R}}_k^*$ with $(c_{\mathcal{R}_k^{(h)},k} - c_{\bar{\mathcal{R}}_k^*,k}) > z_0'$ have been removed from $\mathcal{R}_k^{(h)}$.

$$\begin{aligned} c_{\mathcal{R}_k^*,k} &= c_{\mathcal{R}_k^{(h)},k} - c_{\bar{\mathcal{R}}_k^*,k} \\ &> z_0' \end{aligned}$$

   Removing one or more requests may allow the feasibility of the schedule, however, if the difference between the cost of the original set of requests and the cost of the set of the removed requests is still greater than z_0', as from Lemma 1, z_0' is an upper bound, the new set of requests is also infeasible and does not satisfy the Benders cut.

3. In $\mathcal{R}_k^*$, some requests $\bar{\mathcal{R}}_k^*$ with $c_{\mathcal{R}_k^{(h)},k} - c_{\bar{\mathcal{R}}_k^*,k} \leq z_0'$ have been removed from $\mathcal{R}_k^{(h)}$.

$$\begin{aligned} c_{\mathcal{R}_k^*,k} &= c_{\mathcal{R}_k^{(h)},k} - c_{\bar{\mathcal{R}}_k^*,k} \\ &\leq z_0' \end{aligned}$$

   If the cost of the set of requests removed is less than or equal to the difference between the original cost and the removed cost, the schedule could potentially be feasible and as shown above, the Benders cut is satisfied.

4. $\mathcal{R}_k^*$ and $\mathcal{R}_k^{(h)}$ share some requests (i.e., $\mathcal{R}_k^{(h)} \cap \mathcal{R}_k^* \neq \emptyset$).

$$c_{\mathcal{R}_k^*,k} = c_{\mathcal{R}_k^* \cap \mathcal{R}_k^{(h)},k} + c_{\mathcal{R}_k^* \setminus \mathcal{R}_k^{(h)},k}$$

   Since the requests in $\mathcal{R}_k^* \setminus \mathcal{R}_k^{(h)}$ are not in the set $\mathcal{R}_k^{(h)}$, they are unconstrained. If $c_{\mathcal{R}_k^* \cap \mathcal{R}_k^{(h)},k} > z_0'$, then similar to case 2, the shared requests have a total cost higher than z_0', thus $R_k^{(h)}$ does not have a feasible solution and does not satisfy the Benders cut. Otherwise if $c_{\mathcal{R}_k^* \cap \mathcal{R}_k^{(h)},k} > z_0'$, the new set could potentially be feasible and satisfies the Benders cut.

5. $\mathcal{R}_k^*$ and $\mathcal{R}_k^{(h)}$ do not share any requests. Since the requests in $\mathcal{R}_k^*$ are not in set $\mathcal{R}_k^{(h)}$, thus are unconstrained. The new set could yield feasible solutions and satisfies the Benders cut.

The above holds for all iterations and set of requests. In addition, the above relationships are exhaustive

and demonstrate that a Benders cut is not satisfied only when the solution is infeasible. This concludes the proof for Lemma 2.

□

Theorem 3. *The proposed inequality modelled in Equations* (5.53) *and* (5.54) *is a valid Benders cut.*

Proof. Lemma 1 showed that z'_0 is an upper bound to the cost of a subset of requests assigned to a specific vehicle, thus by applying the cuts iteratively, the algorithm cuts off infeasible solutions and the algorithm will terminate by finding a feasible solution or prove infeasibility. Furthermore, the inequality does not remove any feasible solutions as shown in Lemma 2. We can then conclude that the proposed inequality is a valid Benders cut. □

5.2.4 A Heuristic Approach

The objective of the STPOB is to minimize the total expected cost. In Equation (5.1), there are two components, $(1 - \pi_r) \times B_r$ which is only dependent on the request and V_k which depends on the vehicle assigned. Since the objective is to minimize the total cost, assigning higher cost requests to lower cost vehicles should lower the total cost.

The algorithm sorts all requests in descending order of cost ratio defined as $\frac{weight}{length}$, and all vehicles in ascending order of cost factor, with ties broken by sorting smaller time windows first. The algorithm assigns each request, in order, to the first vehicle that is able to take it. If no vehicle can satisfy the request, it is assigned to the taxi. Once the request is assigned to a vehicle, the vehicle is split into two vehicles with the time window availability before the assigned request and the time window availability after the assigned request. The list of vehicles is then resorted. The algorithm is outlined in Algorithm

2.

Algorithm 2: Construction Heuristic for the STP.

Data: Set of requests $\mathcal{R}$, and set of vehicles $\mathcal{K}$

Result: A set of scheduled routes

1 Sort $\mathcal{R}$ based on descending order of $\frac{weight}{length}$;
2 Sort $\mathcal{K}$ based on ascending order cost factor, break ties with smaller time windows first;
3 **for** *all requests* r *in* $\mathcal{R}$ **do**
4 **for** *all vehicles* v *in* $\mathcal{K}$ **do**
5 **if** r *can be served by* v **then**
6 assign r to v and set start time of r as earliest start time on r that is after the earliest pickup time of r;
7 split v into 2 vehicle pieces, v_1 and v_2 with
(i) start time of v_1 = start time of v
(ii) end time of v_1 = start time of r
(iii) start location of v_1 = start location of v
(iv) end location of v_1 = pickup location of r
(v) start time of v_2 = end time of r
(vi) end time of v_2 = end time of v
(vii) start location of v_2 = delivery location r
(viii) end location of v_2 = end location of v ;
8 insert v_1 and v_2 into $\mathcal{K}$ based on the new time window lengths ;
9 break;
10 **end**
11 **end**
12 **end**
13 Regroup all pieces of the same vehicle to make scheduled routes ;

5.3 Experimental Results

All five approaches have been tested on both the randomly generated dataset and the CHATS dataset described in Chapter 3. An analysis similar to that of Chapter 4 is presented for the STPOB. Comparisons between the performance of the different algorithms on the STPOB and the STP are also drawn. Finally the best approaches, CP, MIP/CP LBBD and the Construction Heuristic are tested on the CHATS dataset.

All five approaches are coded in C++ using IBM's CPLEX Studio 12.7. The experiments are run on a computer with Intel Xeon E3-1226 v3 @ 3.30GHz, 16G RAM using single thread and a 600 second runtime limit for both the generated dataset and the CHATS dataset.

5.3.1 Randomly Generated Dataset Results

Compared to the STP, the STPOB seems to be a harder problem to solve in general. The number of instances solved to optimality in all approaches are less compared to the STP. Figure 5.1 summarizes the experimental results on the generated dataset.

In descending order of best performance, CP solved 73 (97%) instances to optimality in an average of 21.78 seconds, MIP/CP LBBD solved 66 (88%) instances in an average of 28.31 seconds, CP/CP LBBD solved 54 (72%) instances in an average of 45.70 seconds and MIP solved 28 (37%) instances in an average of 56.20 seconds. The solution computed by the construction heuristic is also used to warm start the four exact techniques, denoted with prefix *with Heuristic*. The heuristic start seems to help little for the STPOB especially in terms of runtime though for CP/CP LBBD, the average runtime is longer with the heuristic start.

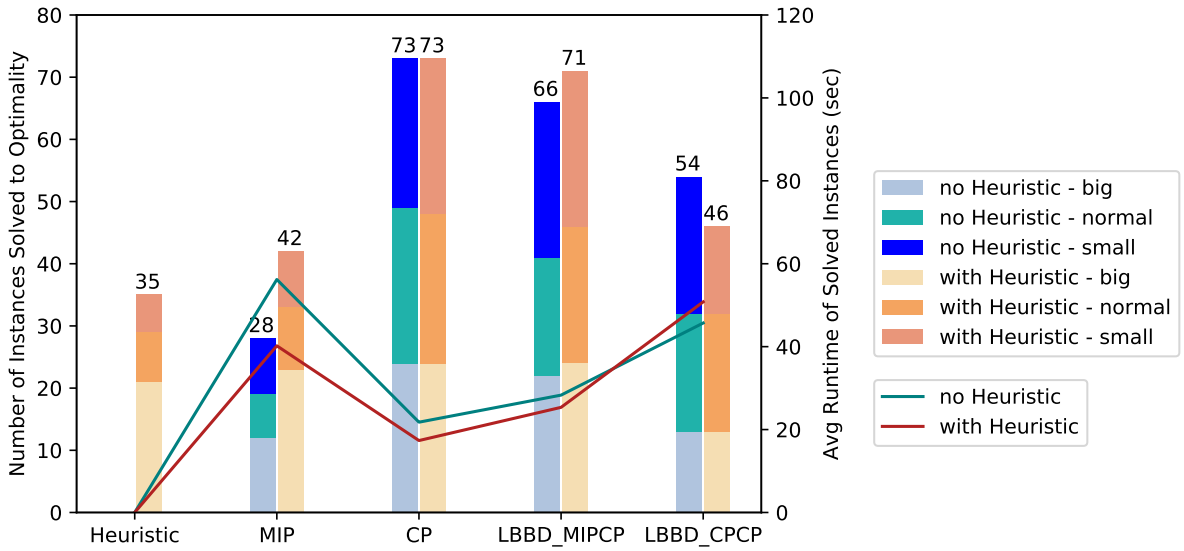


Figure 5.1: Number of instances solved to optimality and average runtime for generated dataset for STPOB.

MIP, MIP/CP LBBD and CP/CP LBBD runtimes are compared to that of the CP approach in Figure 5.2. It can be clearly seen that CP is the best approach. It is also worth noticing that for CP vs. MIP/CP LBBD, the majority of the big time window instances have shorter runtime with the MIP/CP LBBD approach compared to the CP approach, while CP outperforms MIP/CP LBBD for the normal and small time windows due to the strong filtering and propagation rules available for shorter time windows in the CP model.

We next look at the runtime comparison of the four different approaches with and without the heuristic start in Figure 5.3. For the CP approach, the heuristic start helps on some instances and hurts on others, with the average runtime remaining approximately the same. On the other hand, the heuristic start helps the MIP approach solve many instances to optimality, especially those with big time windows. Originally, MIP solved 28 instances to optimality; with the heuristic warm start, it solves 35 instances to optimality. In those cases the heuristic warm start solution provides an optimal solution to the MIP model, thus helping the MIP model solving more instances. From the figure, it can be seen that many blue dots (big time window instances) are in the top left corner meaning that these instances timed out while using the MIP approach but the MIP with the heuristic start was able to solve these instances within the time limit. In terms of the MIP/CP LBBD, the heuristic enabled the approach to find a few more optimal solutions. Finally, the heuristic start worsened the CP/CP LBBD approach especially on the small TW instances.

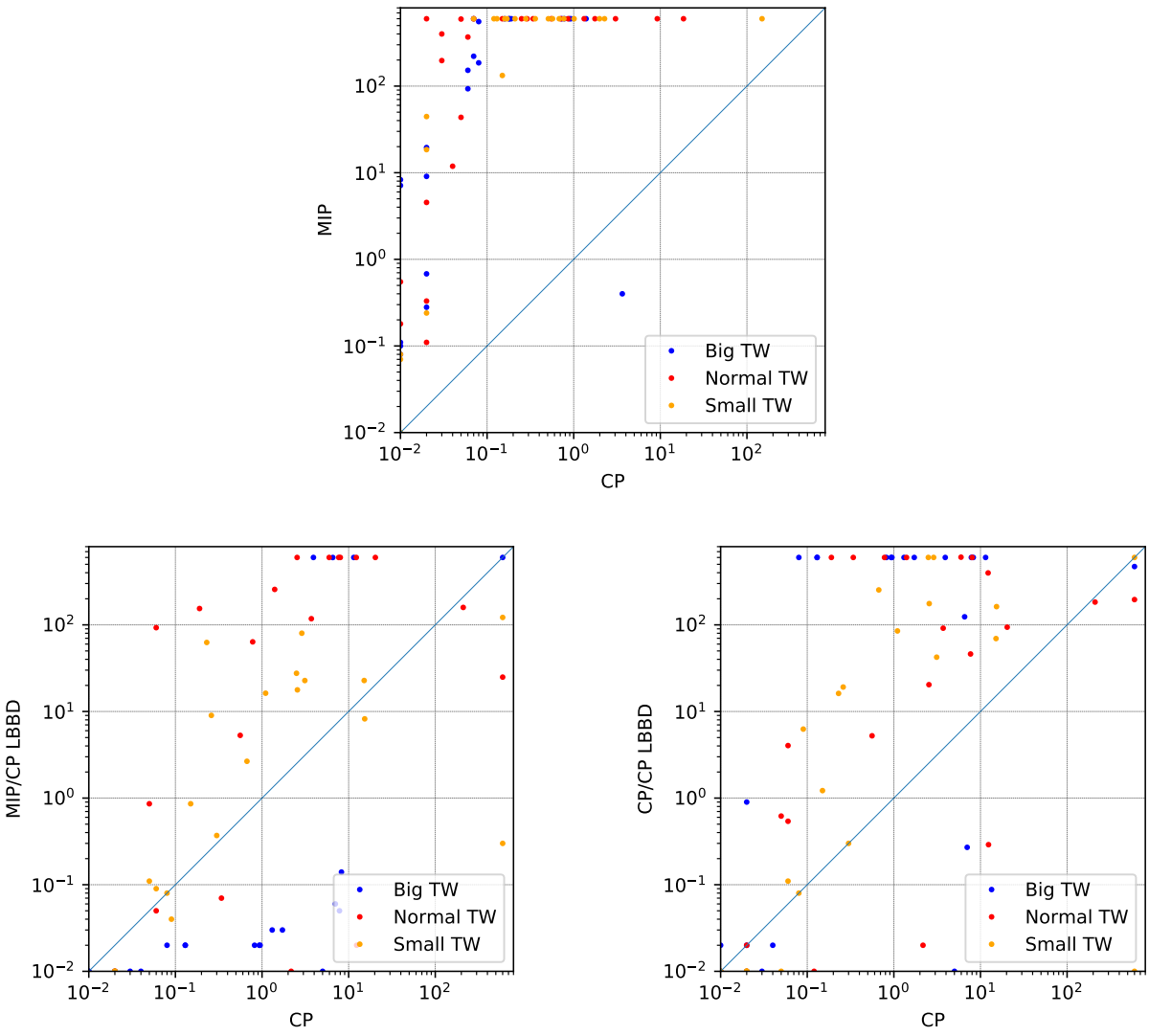


Figure 5.2: Runtime comparison of CP and MIP, MIP/CP LBBD and CP/CP LBBD.

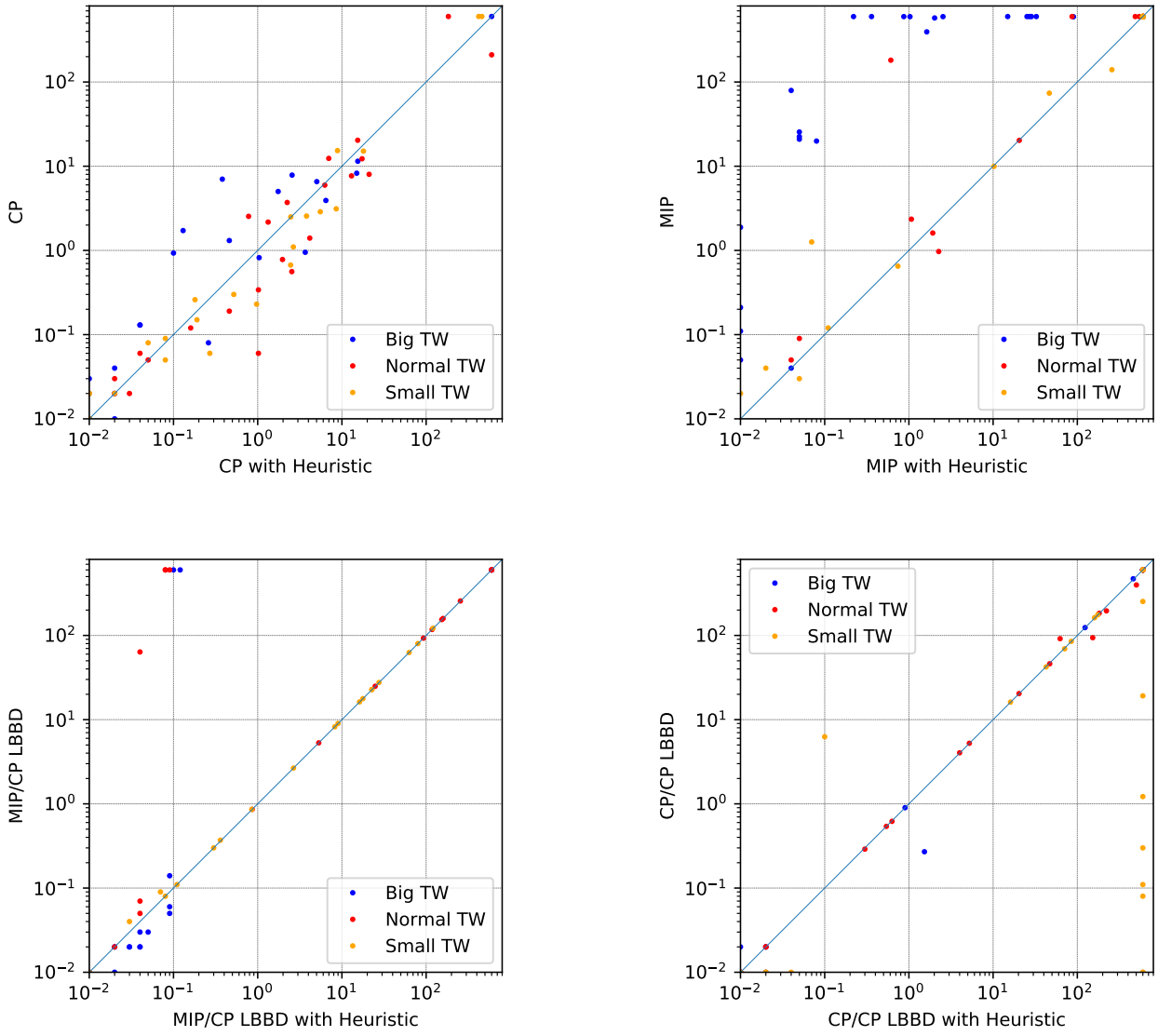


Figure 5.3: Runtime comparison of CP, MIP, MIP/CP LBBD and CP/CP LBBD with and without heuristic start.

The cumulative number of instances solved to optimality over time is seen in Figure 5.4. MIP/CP LBBD seems to dominate in the first 0.1 seconds but subsequently, CP is the dominant approach.

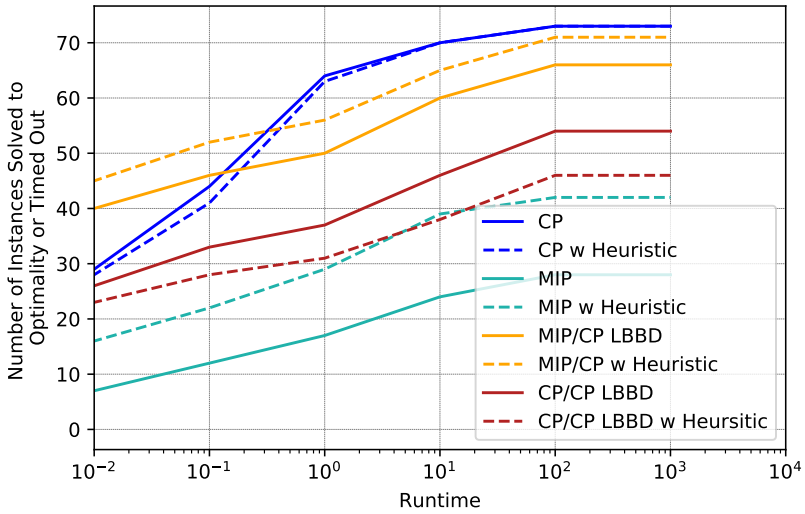


Figure 5.4: Number of instances solved to optimality over total runtime.

Overall, CP is the best performing approach, however, it seems that the two LBBD approaches perform better than when solving the STP. In the next two subsections, we study in detail CP compared to the LBBD approaches.

5.3.2 Analysis of LBBD Approaches

In this subsection, we study the performance of the LBBD approaches. The instances where MIP/CP LBBD failed to find optimal solutions within the time limit seem to be all problems where a particular subproblem is too hard and failed to terminate within the time limit. In contrast, the CP/CP LBBD failed in many hard subproblems but also encountered multiple situations of too many iterations. The proportion of runtime spent in the master problem and the subproblem is shown in Figure 5.5.

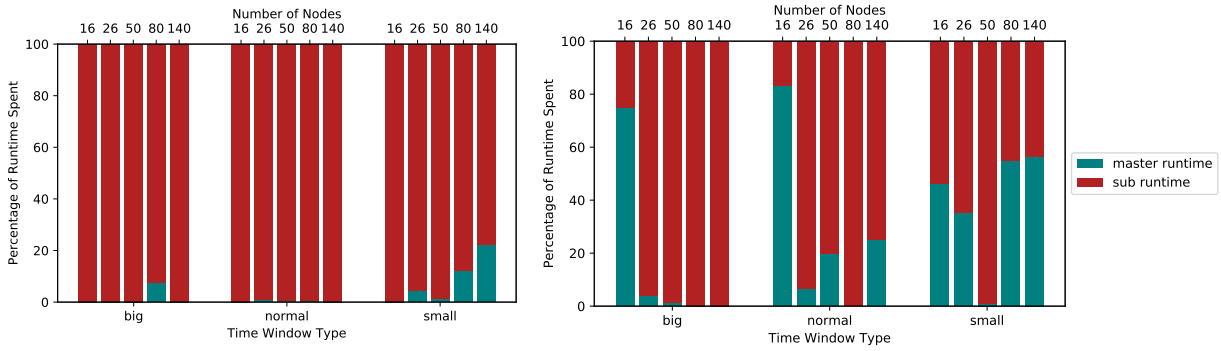
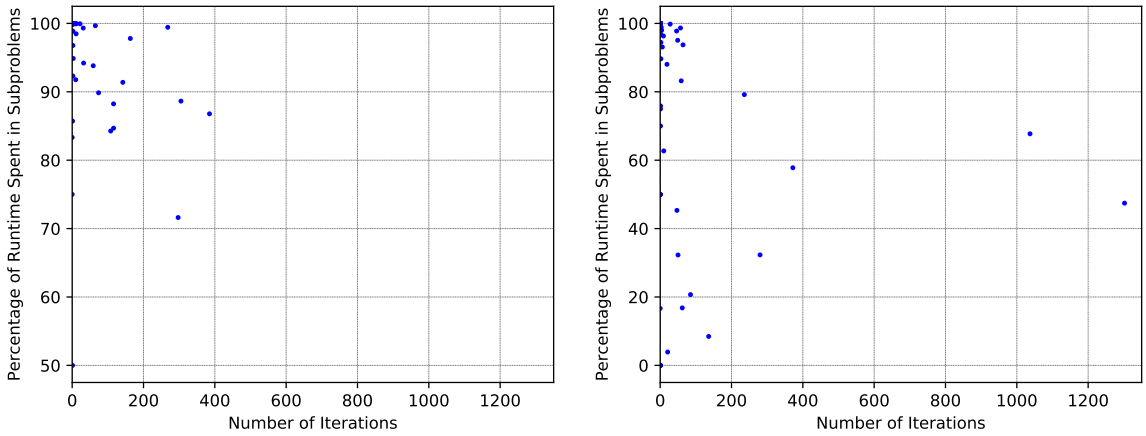


Figure 5.5: Comparison of runtime spent in the master problem vs. the subproblem across all instances.

It can be clearly seen that for the MIP/CP LBBD approach, most of the runtime is spent in the

subproblems. For the CP/CP LBBD, it seems more evenly distributed. However, it is worth noticing that for the 16-node instances, all solved in less than 0.01 seconds spending ∼0 sec in the subproblem and <0.01 sec in the master problem thus resulting in larger percentage of distribution of runtime in the master problem. Again, we observe the same behaviour in the STP where in the LBBD approaches, without all the information, the master problem makes a poor assignment, in particular, creating very hard subproblems where the subproblem cannot be solved within a time limit.

Next, we explore the percentage of time spent in subproblems vs. the number of iterations in each of the MIP/CP LBBD and CP/CP LBBD approaches. As can be seen in Figure 5.6, the majority of the instances are solved in very few iterations.



(a) MIP/CP LBBD.

(b) CP/CP LBBD.

Figure 5.6: Percentage of time spent in the subproblem vs. the number of iterations.

5.3.3 Comparison of the CP Approach and the LBBD Approaches

Again, we study the behaviour of the CP approach compared to the LBBD approaches in terms of their first solution quality and the search space reduction from the first solution found.

CP and LBBD First Solutions

In order to compute the quality of the first solutions, we measure how long it takes to find an initial feasible solution and how good this solution is. The criteria for a first feasible solution remains the same as in the previous chapter (see Section 4.3.3). For each of the instances, we let the CP method run until finding an optimal solution, we then measure the optimality gap of the first solution's objective value for all approaches.

Looking at Figures 5.7 and 5.8, it can be observed that in both LBBD approaches, the optimality gap of the first solution found is much smaller than the one found by the CP approach. In fact, 36 instances were solved to optimality with only one iteration by MIP/CP LBBD and 30 instances with only one iteration by CP/CP LBBD. Therefore, all of these instances have an optimality gap of 0 from the first feasible solution found. However, the time it takes to find a first feasible solution is much smaller for the CP method than the two LBBD methods especially for CP/CP LBBD.

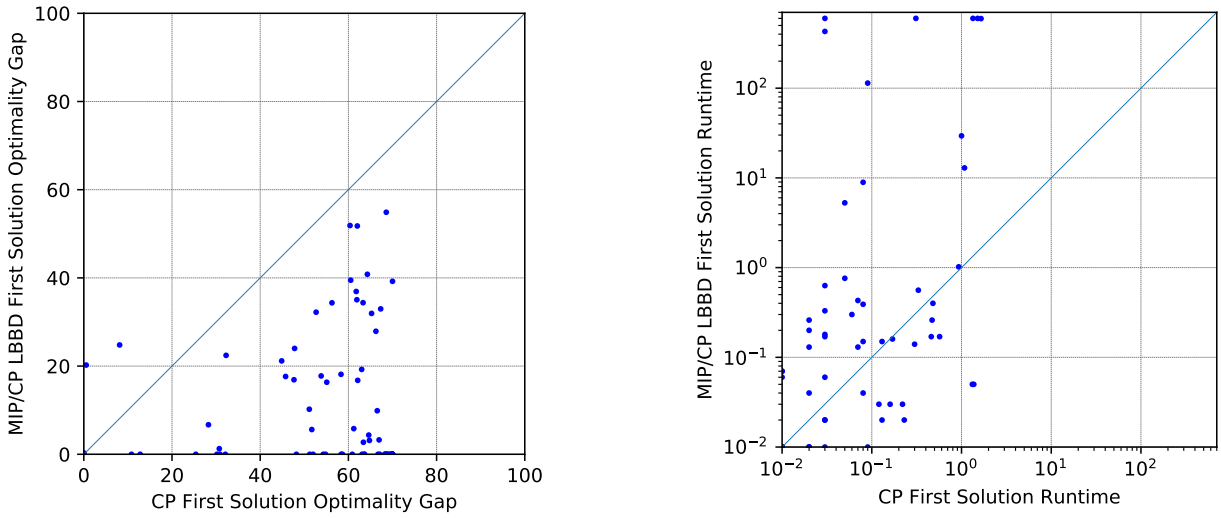


Figure 5.7: First solution quality of MIP/CP LBBD compared to CP.

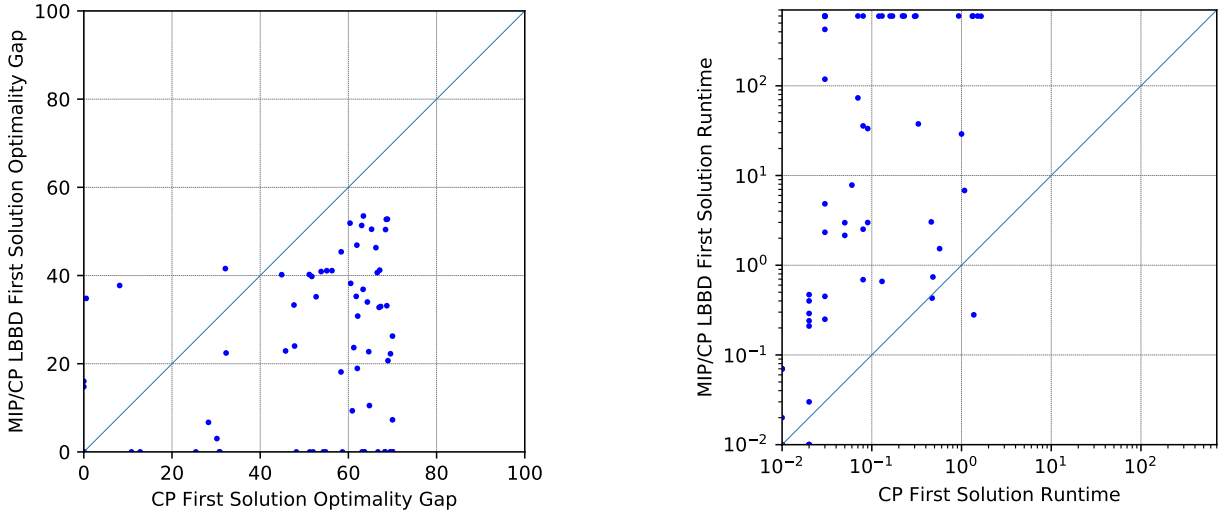


Figure 5.8: First solution quality of CP/CP LBBD compared to CP.

We test the quality of the first solution in both CP and MIP/CP LBBD and its effects on the algorithms through the following experiments. In the first set of experiments, we take the first solution found by the CP approach and use that as a warm start to the MIP/CP LBBD approach. The runtime results can be seen in Figure 5.9. Unlike in the STP, the CP start does not help too much with runtime for the MIP/CP LBBD approach. The number of optimally solved instances decreases to 64 when using the CP starting solution from optimally solving 66 instances without the CP starting solution.

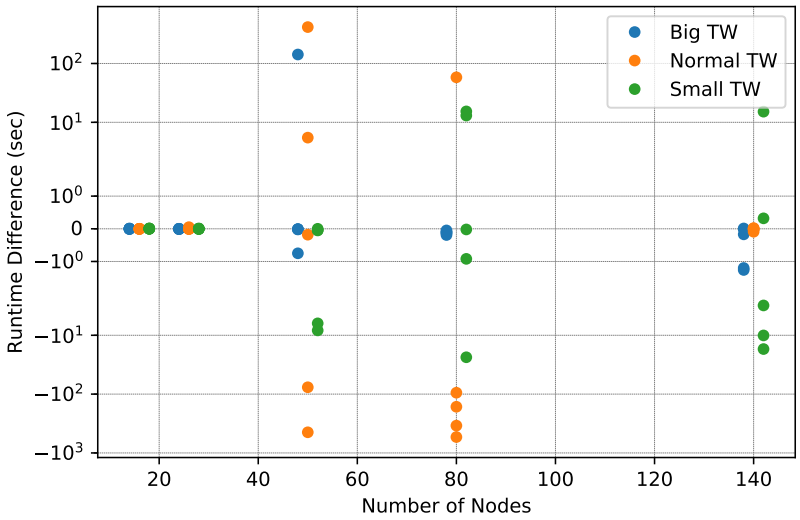


Figure 5.9: Runtime difference of pure MIP/CP LBBD minus MIP/CP LBBD with CP starting solution.

Using MIP/CP LBBD first solution as a warm start solution to the CP approach shows a similar behaviour as the runtime is not affected much by the starting solution and the number of instances solved to optimality with the MIP/CP LBBD is reduced to 72 from 73 without using the MIP/CP LBBD start solution.

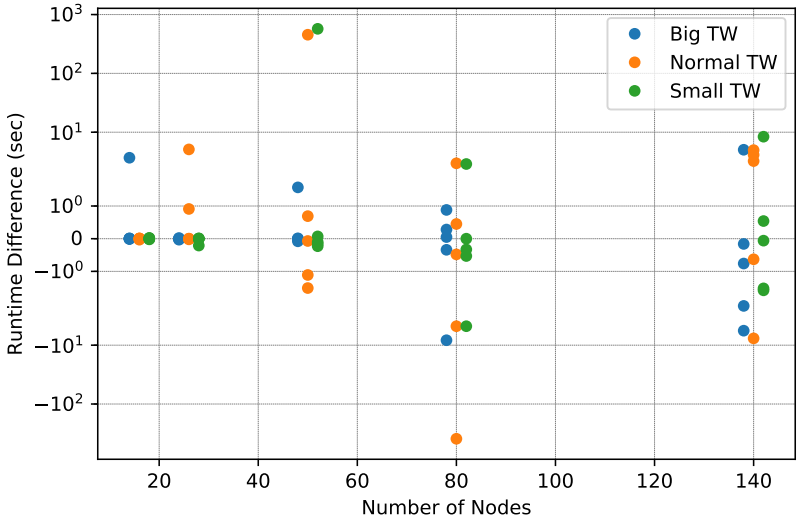


Figure 5.10: Runtime difference of pure CP minus CP with MIP/CP LBBD starting solution.

Although the MIP/CP LBBD provides a better first solution, the amount of time to find the solution is too large to speed up the CP approach. If we let CP run until MIP/CP LBBD finds its first feasible solution, then the quality of the solution found by CP is much better as can be seen in Figure 5.11. In fact, 39 instances are solved to optimality given the time the MIP/CP LBBD used to find the first solution. It is also worth noticing that for 11 instances, CP failed to find any feasible solution in the given runtime.

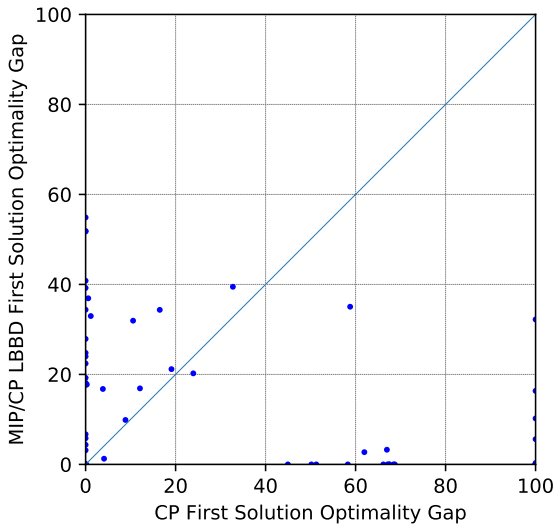


Figure 5.11: Comparison of CP vs. MIP/CP LBBD solution quality given runtime of finding MIP/CP LBBD first solution.

Search Space Reduction

Since the STPOB is a minimization problem, in order to measure the impact on the search space reduction, we provide artificial upper bounds. The search space is computed as described in the previous chapter (see Section 4.3.3).

We let the CP approach run until finding an optimal solution and compute 6 different upper bounds that are 100%, 120%, 140%, 160%, 180%, and 200% of the optimal objective value. This upper bound is added as a new constraint and the algorithms are then allowed to simply propagate without branching. We compute the size of the search space of the original problem and with the additional upper bound constraint and observe any reduction in search space. The results are presented in Table 5.3 for CP and Table 5.4 for CP/CP LBBD. For each time window type, there are 25 instances and the number indicates how many instances show a decrease in the size of the search space with the addition of an artificial upper bound.

From the observations, it seems that the majority of the instances show a reduction in search space size when given an extremely tight upper bound, thus finding a good quality feasible solution is crucial. In contrast, instances show more space reduction for the CP/CP LBBD approach which confirm the better performance of LBBD approaches solving the STPOB compared to the STP (see Section 4.3.3).

Table 5.3: Number of instances that show a reduction in search space after applying an artificial upper bound for the CP model.

CP		Upper Bound Percentage					
		100%	120%	140%	160%	180%	200%
Time Window Type	small	18	1	0	0	0	0
	normal	22	2	2	1	0	0
	big	23	0	0	0	0	0

Table 5.4: Number of instances that show a reduction in search space after applying an artificial upper bound for the CP/CP LBBD approach.

CP/CP LBBD		Upper Bound Percentage					
		100%	120%	140%	160%	180%	200%
Time Window Type	small	23	7	1	0	0	0
	normal	25	12	7	4	0	0
	big	24	13	9	3	0	0

5.3.4 Comparison of STP and STPOB Solutions

In an attempt to understand the different objectives of the STP and the STPOB and the impact they may have on problem difficulty, we investigated the solution structures of the same instances solved using CP. The objective of the STP is to maximize the number of requests fulfilled given a set of vehicles, while the STPOB aims to minimize the total cost of assigning all requests with additional taxi vehicles. All instances are kept the same (i.e., all vehicle and request information are identical) except for the addition of vehicle cost and cancellation probability of requests when solving the STPOB.

First, we took the solutions computed to solve the STP and compute their costs in the same way as in the STPOB; each request uses the same probability of cancellation and each vehicle has the same cost factor. Note that when computing the optimal solutions for STP, vehicle cost factors were not part of the objective function and thus requests were assigned to vehicles arbitrarily. Figure 5.12 highlights the cost of each instance solved as an STP compared to those solved as an STPOB. The blue datapoints represent the instances where all requests were fulfilled in the STP solutions, while the orange datapoints represent the instances where only a subset of requests were fulfilled. We can see that only the instances where all requests could be scheduled incurred a higher cost in the STP. This is due to the arbitrary assignment of vehicles since there are no cost differences in the STP. The rest of the instances incurred a lower cost since in the STP, as there is no taxi vehicle and the requests that are not scheduled do not incur any cost.

We then examined the number of vehicles used for each instance and the variance in the number of requests assigned to each vehicle. Note that taxi vehicles are not counted. In Figure 5.13, we can see in almost all instances, more vehicles are employed in STP solutions. In these solutions, where vehicles do not have a cost, the organization vehicles and the volunteer vehicles are not distinguished, whereas in the STPOB solutions, the vehicles have different cost, thus most requests are assigned to volunteer

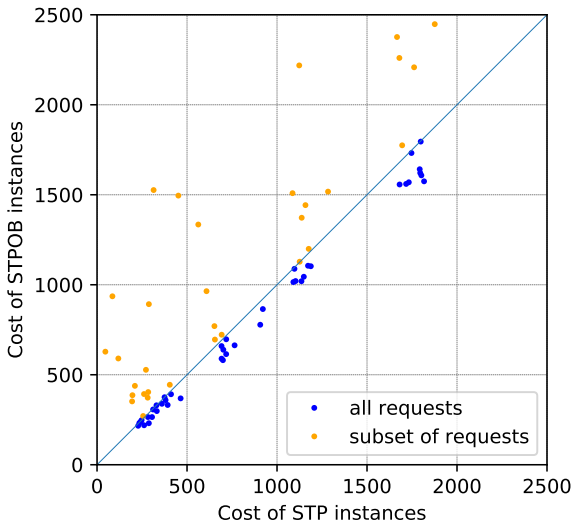


Figure 5.12: Cost comparison of instances solved in STP vs. STPOB.

vehicles. Additionally, from Figure 5.14, we can see that the variance in the number of requests assigned to each vehicle is a lot higher for the STPOB, indicating that in the STPOB, fewer vehicles are loaded and some vehicles are much more heavily employed than others when compared to solving the STP. This also suggests that requests are more evenly spread to each vehicle in STP solutions compared to the STPOB solutions.

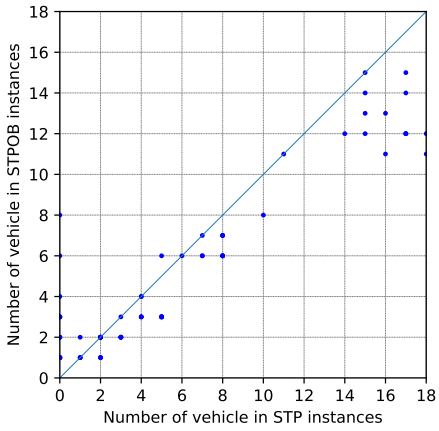


Figure 5.13: Comparison of number of vehicles used in solutions of instances solved in STP vs. STPOB.

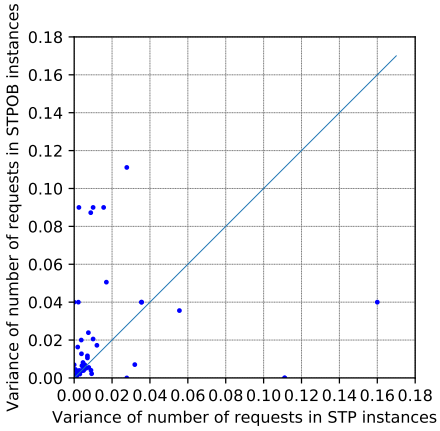


Figure 5.14: Comparison of variance in the number of requests per vehicle of instances solved in STP vs. STPOB.

5.3.5 CHATS Dataset Results

Based on the experimental results from the previous subsections, again, the CP and MIP/CP approaches seem to be the best performing methods. In this subsection, we observe the performance of the CP and MIP/CP LBBD methods on the CHATS dataset.

As expected, the STPOB is a harder problem compared to the STP, thus out of 280 instances, CP was only able to solve 93 instances to optimality in an average runtime of 284.34 seconds while MIP/CP LBBD only solved 1 instance to optimality. The Heuristic method had an average runtime of 112.11 but it is not possible to assess which instances have reached optimality. By examining the runtime of the instances solved by the CP approach, it can be clearly seen that as the number of nodes increase, the problem gets harder as shown in Figure 5.15.

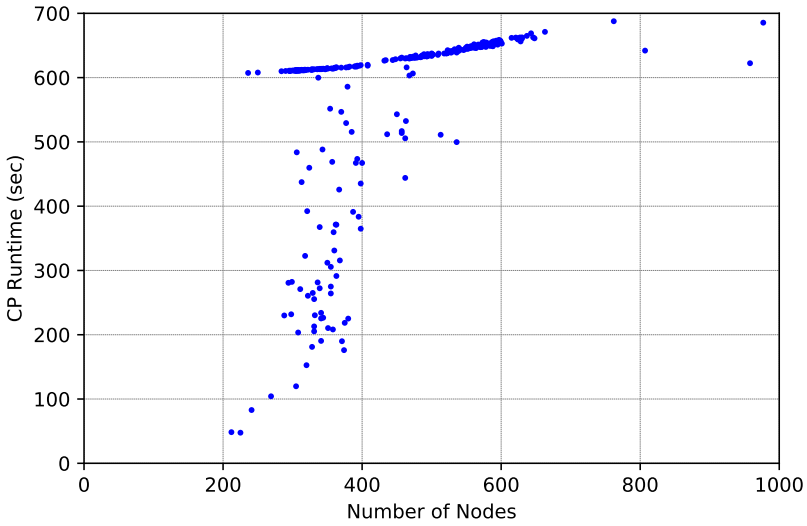


Figure 5.15: CP runtime of CHATS instances over number of nodes.

The heuristic approach was also run on the CHATS dataset, and we compared both the objective value and the runtime of CP against the Heuristic approach as well as MIP/CP LBBD. In Figure 5.16, it can be seen that in the runtime for MIP/CP LBBD, all instances timed out except for one and the quality of the solution for the CP approach dominates MIP/CP LBBD in all of the instances. In both runtime and objective value, the smaller the number the better, thus all instances above the diagonal line are instances where CP performs better than MIP/CP LBBD.

We compare CP to the heuristic approach in Figure 5.17. Even though the heuristic runtime is better in all instances, the quality of the solution is much poorer than the CP approach. On average, the CP approach has a cost that is 633.39 less than the Heuristic approach.

5.4 Conclusions

In this chapter, we defined the Senior Transportation Problem with Overbooking (STPOB) which is an extension of the STP presented in the previous chapter as we see an opportunity to take extra requests based on the probability of cancellation that is associated with each request. The biggest difference between the STPOB and the STP is the change in the objective function and the addition of vehicle cost factors that differentiate the vehicles. In the STP, the vehicle heterogeneity only contributes to the constraints. In the STPOB, the assignment of a request to a different vehicle could result in different costs which makes the problem harder to solve. On the other hand, all of the route constraints are the same as for the STP except for the creation of the extra taxi vehicle, which is unconstrained and

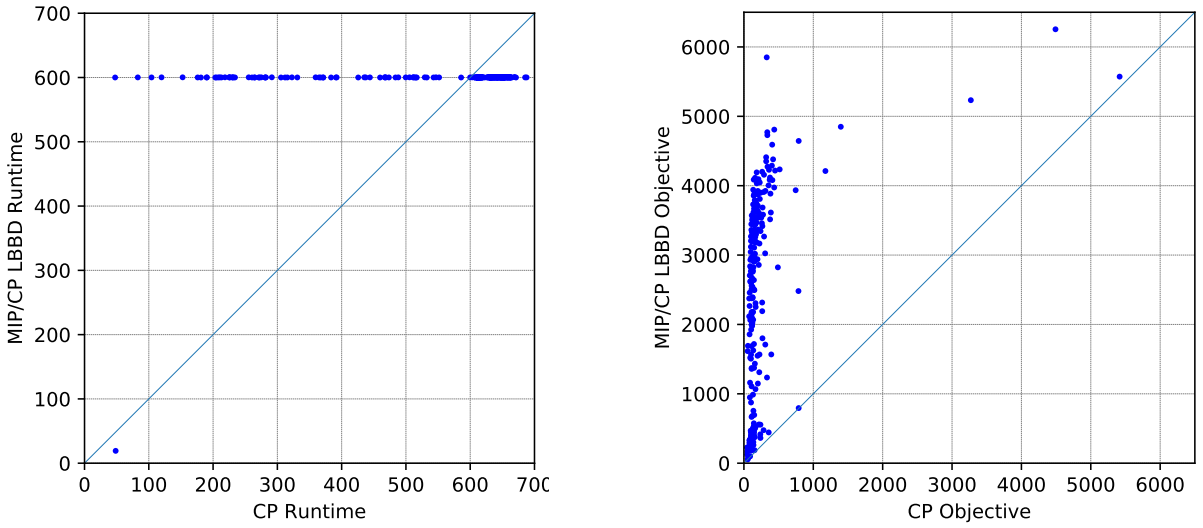


Figure 5.16: Objective value and runtime comparison between CP and MIP/CP LBBD.

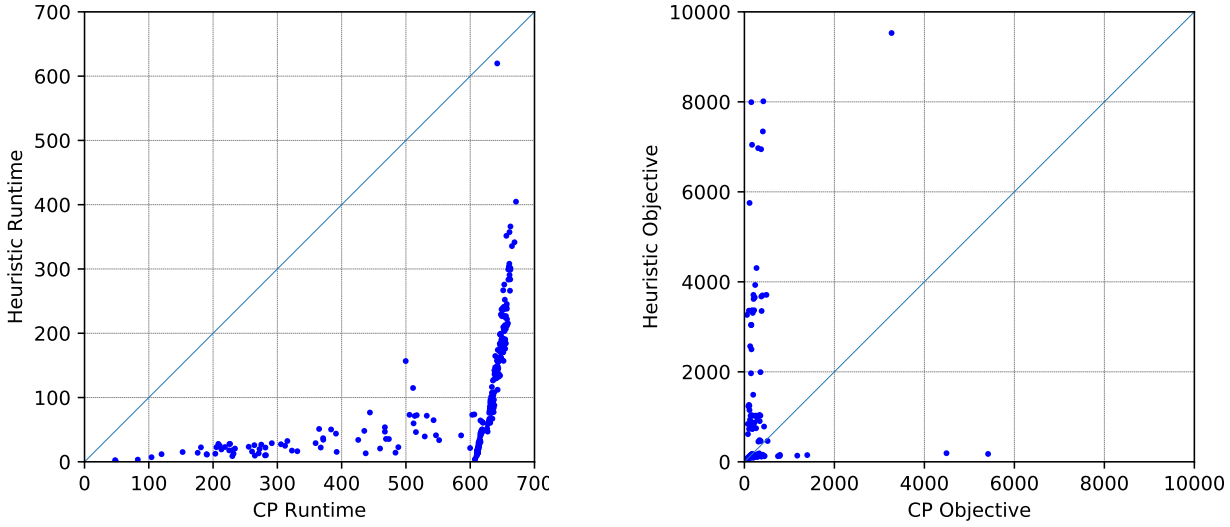


Figure 5.17: Objective value and runtime comparison between CP and the heuristic approach.

assumed to be able to fulfil any request.

Based on the experimental results, the STPOB is harder to solve overall and a few instances were not solved to optimality. The CP approach performs the best followed by MIP/CP LBBD, CP/CP LBBD, then MIP. However, the LBBD approaches seem to perform better in terms of solving instances to optimality and runtime compared to the STP. The good performance is supported by the large search space reduction abilities of this model structure. Nonetheless, the large amount of time required to compute a first solution for the LBBD approaches hinders their runtime in general. Furthermore, the number of iterations for the MIP/CP LBBD approach seems to have decreased compared to the number of iterations for the STP's MIP/CP LBBD and all the instances that failed to solve to optimality were due to a hard subproblem.

For the CHATS instances, again CP dominates MIP/CP LBBD where CP was able to solve 93 instances whereas MIP/CP LBBD only solved 1 instance in the given 600 seconds time limit. Overall, the STPOB is a harder problem to solve compared to the STP as the different vehicle assignment contributes differently to the objective function but the CP model still shows promising results in terms of solution quality and runtime.

Chapter 6

Conclusions and Future Work

In this chapter, we conclude this thesis with a summary of the work presented in previous chapters and present the contributions of this research work. We then close with directions for potential future work.

6.1 Summary and Contributions

This thesis has presented the formal definitions of two novel optimization problems: the Senior Transportation Problem (STP) and the Senior Transportation Problem with Overbooking (STPOB). Both of these problems are inspired by a real-life problem faced by non-profit organizations that provide senior transportation services. The STP maximizes the weighted sum of requests served, while the STPOB minimizes the total expected costs while allowing overbooking of requests. One contribution of this work is to bring attention to these two problems that are variations of Pickup and Delivery Problems with Time Windows (PDPTW) but with distinct objective functions and additional constraints specific to senior transportation.

We randomly generated 75 problem instances, varying problem sizes and time windows of each request and vehicle, and extracted 280 instances from the CHATS dataset for each of the STP and the STPOB. We performed analysis on the CHATS dataset and observed some seasonality trends in both the demand (number of requests) and supply (number of available vehicles and drivers). From the CHATS dataset, we identified 9 features: age, gender, language preference of the client, weekday, month, weight, distance, fare and size of the request to use towards building a model to predict the probability of cancellation of a given request. This probability is then used to generate STPOB problem instances.

To solve the STP, we provided five solution approaches: Mixed Integer Programming (MIP), Constraint Programming (CP), two Logic-based Benders Decompositions, one with a MIP master problem (MIP/CP LBBD) and one with a CP master problem (CP/CP LBBD), and one construction heuristic. On the generated dataset, CP solved the most number of instances to optimality followed by MIP/CP LBBD, CP/CP LBBD, the construction heuristic, and MIP, where except for the construction heuristic, all other approaches also proved optimality. Out of the instances that were not solved to optimality by the LBBD approaches, we observed two phenomena, either the algorithms time-out solving a very hard subproblem or they perform many iterations. When observing the behaviour of the CP approach, we saw that CP is able to find good quality first solutions and that the lower bound provided from the first solutions has a greater impact on the search space reduction compared to the LBBD approaches.

Compared to CP, LBBD algorithms find better first solutions but at a higher computation time and those solutions have less impact on search space reduction.

We also used the construction heuristic to compute a warm start solution for the four exact algorithms and observed the resulting performance. The heuristic improved the MIP approach significantly both in terms of the number of optimal solutions found and the average runtime, slightly improved the two LBBD algorithms, but did not affect the CP approach. The two best algorithms, CP and MIP/CP LBBD were then tested on the CHATS dataset. Again, CP shows the best performance and was able to solve 237 out 280 instances to optimality within 600 seconds.

We took similar approaches to the STPOB, adapting our five solution techniques for the STP. The STPOB proved to be a harder problem to solve than the STP. Given the same problem instances that were used for the STP with added cancellation probability, all five algorithms solved fewer instances to optimality with a higher average runtime for the STPOB. Nevertheless, CP still demonstrated the best performance followed by MIP/CP LBBD, CP/CP LBBD, the construction heuristic, and MIP. The heuristic was also used to compute a warm start for the four exact algorithms. Unlike for the STP, the heuristic warm start did not help the other algorithms very much and even worsened the performance of CP/CP LBBD.

For the two LBBD approaches, the number of iterations for all instances decreased compared to the approaches without the heuristic warm start. There were no instances where MIP/CP LBBD fails due to too many iterations; all instances failed due to hard subproblems. The search space reduction experiments showed different results than for the STP: the artificial bound on the solution quality has a greater impact on the search space using CP/CP LBBD than using CP. However, CP finds good first solutions faster and still dominates the LBBD approaches in terms of performance. CP and MIP/CP LBBD were tested on the CHATS instances and CP solved 93 instances to optimality while MIP/CP LBBD only solved one instance to optimality.

6.2 Future Work

In the STP, we considered one deterministic optimization problem, however, when assigning requests, a fair selection of requests to fulfil could also be considered. For example, if a trip to a grocery store has a low weight, then when resources are limited, perhaps no grocery store visits will ever get scheduled. The percentage of requests fulfilled based on request types (i.e., grocery runs, medical appointment, etc.) may be added as an extra factor to the objective function such that there is an equal spread amongst the types of satisfied requests. Furthermore, if the problem is non-uniformly spread geographically, requests from clients living in isolated locations will often not be able to be satisfied. Therefore, the consideration of a fair and balanced representation of all request types is also important.

We employed a three-indexed MIP model, while Furtado et al. proposed a two-indexed formulation for the PDPTW which they claim to have better performance than the common three-indexed formulation [24]. Studying how to modify Furtado's formulation to fit the STP and evaluating its performance is a future next step.

For the STPOB, the objective function is an estimation of the true cost as all costs have been converted to be based on travel time. However there are also components such as gas expenses that are truly based on the travel distance that were not explicitly represented. In future study, we can provide problem formulations based on true costs.

Furthermore, we have not considered the problem of rescheduling a request when a cancellation happens. In the STPOB, when a request is cancelled, the vehicle is idle for that period of time and no travel cost occurs. Therefore, exploring methods to create schedules such that it is easy to re-schedule, given a cancellation is a future research interest. One research direction is the notion of slot compression [59] from the clinical appointment scheduling literature to create a two-dimensional compression from both temporal and capacity constraints to allow flexible start and end times of trips.

In both Chapters 4 and 5, we performed experiments to understand why one approach performs better than another. We explored two concepts: the quality of first solution found and its impact on the search space reduction. However, there are still important open questions about why CP performed so well. Some directions to explore include analyzing the impact of the default CP Optimizer search which concurrently run Large-Neighbourhood Search and Failure Directed Search on the way the different decisions are made and interleaved during the search. Through these further analyses, we hope to generalize from these instances to discover what aspects contribute to CP's success. Other experiments such as analyzing improvement of the LBBD objective value at each iteration could help us better understand the behaviour of the algorithms on different problems. These tests could be generalized to assess specific problem characteristics and constraints. Furthermore in both chapters, the two LBBD approaches struggle on hard subproblems. Thus, with further tests to understand what circumstances create a hard subproblem and if there are ways to avoid these situations, the two LBBD approaches could be improved and alternative LBBD's can be explored.

6.3 Conclusion

The goal of this thesis is to present two novel problems, the STP and STPOB to the Operations Research community. We believe that the STP and STPOB have very interesting problem characteristics that can be further studied. We presented five different approaches to solve each of the problems and performed empirical analysis on both a randomly generated dataset and a real-world dataset. We demonstrated that CP is a good approach to solve both the STP and STPOB even for very large instances.

Appendices

Appendix A

STP and STPOB Parameters

Table A.1: STP parameters.

Parameter	Description
G	Graph representing the locations of vehicle depots and clients
$\mathcal{V}$	Set of vertices
$\mathcal{A}$	Set of arcs
$T_{i,j}$	Travel time between two location vertices in $\mathcal{V}$. All entries are strictly non-negative. A large entry indicates that no route is allowed between two locations. $T_{i,i} = 0, \forall i \in \mathcal{V}$ and $T_{i,j}$ does not have to equal $T_{j,i}$, tt represents the travel time matrix
$\mathcal{D}$	Set of depot location vertices, $\mathcal{D} \subset \mathcal{V}$
$\mathcal{K}$	Set of vehicle
$\lvert\mathcal{K}\rvert$	Number of available vehicles
$\mathcal{N}$	Set of client location vertices, $\mathcal{N} \subset \mathcal{V}$
$\mathcal{R}$	Set of requests
$\lvert\mathcal{R}\rvert$	Number of requests
E_i	Earliest service start time at each location in $\mathcal{V}$
L_i	Latest service start time at each location in $\mathcal{V}$
S_i	Service time needed to perform the task at each location in $\mathcal{V}$.
Q_r	Load size associated with each request r in $\mathcal{R}$
W_r	Weight associated with each request r in $\mathcal{R}$. All entries are strictly non-negative.
C_k	Capacity of each vehicle k in $\mathcal{K}$
Z	Maximum time horizon
F	Maximum ride time of a client

Table A.2: Additional parameters for the STPOB.

Parameter	Description
π_r	Probability of cancellation of request r
B_r	Total travel time length of request r. $B_r = T_{i_{r^+}, i_{r^-}}$
V_k	Cost factor of vehicle k. $V_{taxi} > V_{paiddriver} > V_{volunteer}$
O_r	Expected cost a request r

Appendix B

Experimental Results for STP

The tables in this appendix present the details of the experimental results on the generated dataset. The tables are organized by the different time windows: big, normal, and small. The columns present the objective (obj), runtime in seconds, and number of iterations (itrs) of the five different approaches including the construction heuristic method, Mixed Integer Programming (MIP), Constraint Programming (CP), logic-based Benders decomposition with MIP master problem and CP subproblem (MIP/CP LBBD), and LBBD with CP master problem and CP subproblem (CP/CP LBBD) without and with the heuristic start (indicated with "_H" after the name of the approach). Grey cells indicate instances that have run out of time without finding and proving an optimal solution.

Big Time Windows									
# of	Heuristic	CP		CP_H		MIP		MIP_H	
Nodes	obj	obj	runtime	obj	runtime	obj	runtime	obj	runtime
	21	21	0.01	21	0.00	21	0.11	21	0.02
	21	21	0.01	21	0.01	21	0.10	21	0.02
16	21	21	0.01	21	0.01	21	0.08	21	0.01
	16	16	3.64	16	1.39	16	0.40	16	0.01
	21	21	0.02	21	0.01	21	0.28	21	0.02
	37	37	0.01	37	0.02	37	7.12	37	0.05
	37	37	0.02	37	0.01	37	19.53	37	0.06
26	37	37	0.01	37	0.03	37	8.30	37	0.04
	37	37	0.02	37	0.02	37	9.07	37	0.04
	37	37	0.02	37	0.02	37	0.68	37	0.04
	57	57	0.07	57	0.15	57	221.09	57	0.24
	56	57	0.06	57	0.14	57	93.15	57	124.22
50	57	57	0.08	57	0.16	57	555.40	57	0.50
	57	57	0.06	57	0.14	57	152.18	57	0.32
	57	57	0.08	57	0.16	57	185.67	57	0.24
	86	86	0.18	86	0.36	24	598.99	86	1.50
	86	86	0.18	86	0.44	19	599.08	86	0.98
80	86	86	0.07	86	0.47	44	599.07	86	2.10
	86	86	0.19	86	0.44	83	599.09	86	1.77
	86	86	0.16	86	0.44	25	599.04	86	0.91

	153	153	0.89	153	1.85	0	599.43	153	22.23
	153	153	0.73	153	1.75	0	599.44	153	18.02
140	153	153	0.78	153	2.82	0	599.45	153	18.04
	153	153	1.39	153	2.80	0	599.43	153	11.29
	153	153	0.93	153	2.03	0	599.43	153	18.62

# of	MIP/CP			MIP/CP_H			CP/CP			CP/CP_H		
Nodes	obj	runtime	itrs	obj	runtime	itrs	obj	runtime	itrs	obj	runtime	itrs
	21	0.00	1	21	0.00	1	21	0.01	1	21	0.00	1
	21	0.00	1	21	0.00	1	21	0.01	1	21	0.00	1
16	21	0.00	1	21	0.00	1	21	0.00	1	21	0.00	1
	16	0.00	1	16	0.00	1	16	0.00	1	16	0.00	1
	21	0.00	1	21	0.00	1	21	0.01	1	21	0.00	1
	37	0.01	1	37	0.00	1	37	0.00	1	37	0.01	1
	37	0.00	1	37	0.00	1	37	0.01	1	37	0.01	1
26	37	0.01	1	37	0.00	1	37	0.01	1	37	0.00	1
	37	0.01	1	37	0.00	1	37	0.01	1	37	0.01	1
	37	0.00	1	37	0.01	1	37	0.01	1	37	0.01	1
	57	0	1	57	0.01	1	48	596.46	2	57	0.02	1
	57	0.01	1	57	0.01	1	57	598.48	2	57	597.87	2
50	57	0.01	1	57	0.01	1	56	596.15	2	57	0.03	1
	57	116.90	1	57	0.01	1	57	604.44	2	57	0.02	1
	57	0.03	1	57	0.01	1	56	595.56	2	57	0.02	1
	86	0.03	1	86	0.02	1	29	615.94	2	86	0.07	1
	86	0.02	1	86	0.02	1	86	598.75	2	86	0.05	1
80	86	0.02	1	86	0.02	1	75	596.15	2	86	0.05	1
	86	0.02	1	86	0.02	1	52	597.04	2	86	0.04	1
	86	0.02	1	86	0.02	1	72	596.32	2	86	0.04	1
	153	340.00	1	153	0.08	1	90	597.14	2	153	0.18	1
	153	2.63	1	153	0.07	1	67	597.37	2	153	0.17	1
140	153	0.04	1	153	0.07	1	48	597.07	2	153	0.17	1
	149	600.01	2	153	0.07	1	47	596.51	2	153	0.17	1
	153	0.04	1	153	0.06	1	45	596.06	2	153	0.17	1

Normal Time Windows									
# of	Heuristic	CP		CP_H		MIP		MIP_H	
Nodes	obj	obj	runtime	obj	runtime	obj	runtime	obj	runtime
	19	21	0.02	21	0.00	21	0.11	21	0.09
	21	21	0.01	21	0.01	21	0.55	21	0.02
16	14	21	0.04	21	0.10	21	11.88	21	9.49
	21	21	0.01	21	0.02	21	0.18	21	0.02
	21	21	0.02	21	0.01	21	0.33	21	0.02

	35	37	0.03	37	0.00	37	400.00	37	14.72
	37	37	0.05	37	0.11	37	43.59	37	0.08
26	33	36	0.02	36	0.02	36	599.23	33	599.13
	27	32	18.52	32	7.12	32	599.24	32	599.09
	36	37	0.02	37	0.03	37	4.54	37	26.54
	51	57	0.06	57	0.08	57	368.52	57	227.59
	56	57	0.05	57	0.12	57	594.31	57	318.63
50	57	57	0.03	57	0.12	57	197.05	57	0.61
	52	57	0.29	57	1.05	54	599.01	55	598.90
	54	54	0.07	54	0.14	54	598.85	54	598.80
	82	86	0.15	86	0.33	53	599.08	82	598.96
	86	86	0.34	86	0.40	46	599.10	82	598.99
80	86	86	0.71	86	1.65	61	599.11	86	1.79
	86	86	3.04	86	0.38	2	599.09	86	1.66
	79	86	0.25	86	0.67	52	599.09	79	599.06
	153	153	9.22	153	6.76	0	599.41	0	599.31
	153	153	1.76	153	3.16	0	599.55	153	26.65
140	153	153	0.55	153	1.21	0	599.47	153	15.58
	153	153	1.32	153	1.80	0	599.53	153	19.95
	153	153	0.86	153	4.07	0	599.51	0	599.32

# of	MIP/CP			MIP/CP_H			CP/CP			CP/CP_H		
Nodes	obj	runtime	itrs	obj	runtime	itrs	obj	runtime	itrs	obj	runtime	itrs
	21	0.00	1	21	0.01	1	21	0.00	1	21	0.01	1
	21	0.03	2	21	0.00	1	21	0.00	1	21	0.00	1
16	21	0.04	1	21	0.03	1	21	0.03	1	20	0.00	1
	21	0.01	1	21	0.00	1	21	0.01	1	16	0.01	1
	21	0.00	1	21	0.00	1	21	0.00	1	21	0.00	1
	37	0.01	1	37	0.01	1	37	0.01	1	37	0.01	1
	37	0.01	2	37	0.00	1	37	0.01	1	37	0.01	1
26	36	0.01	1	36	0.01	1	36	0.87	115	36	0.01	1
	32	0.03	3	32	0.03	3	32	0.14	2	32	0.14	2
	37	0.11	2	37	0.11	3	37	0.11	2	37	0.11	12
	57	1.33	5	57	1.34	5	57	223.03	35	54	371.05	39
	57	0.74	5	57	0.74	5	57	32.18	16	57	14.90	6
50	57	5.00	2	57	0.01	1	57	0.02	1	54	12.35	6
	57	1.70	19	57	1.68	19	57	80.12	107	52	90.41	11
	54	2.63	5	54	0.01	1	54	326.06	20	54	0.04	1
	86	11.63	7	86	11.65	7	57	614.05	4	54	614.82	14
	86	2.61	14	86	2.50	16	68	596.35	2	65	608.01	2
80	86	8.47	13	86	0.03	1	27	601.23	278	27	602.39	2
	86	1.89	9	86	0.02	1	57	620.32	24	43	617.19	11
	86	0.02	1	86	0.02	1	56	594.86	11	63	614.97	4

	153	24.86	58	153	32.52	94	92	608.34	5	79	609.07	2
	153	521.32	13	153	0.09	1	65	601.09	5	102	603.47	1
140	150	600.00	11	153	0.08	1	103	617.21	9	110	614.01	1
	153	599.82	5	153	0.08	1	73	596.77	2	56	604.55	2
	148	600.01	3	148	600.00	4	75	645.05	2	101	612.11	2

Small Time Windows

# of	Heuristic	CP		CP_H		MIP		MIP_H	
Nodes	obj	obj	runtime	obj	runtime	obj	runtime	obj	runtime
	5	5	0.01	5	0.01	5	0.01	5	0.01
	5	5	0.01	5	0.01	5	0.07	5	0.08
16	14	16	0.00	16	0.01	16	2.21	16	1.58
	21	21	0.01	21	0.01	21	0.08	21	0.02
	19	21	0.02	21	0.02	21	0.24	21	0.99
	5	5	0.00	5	0.01	5	77.51	5	42.42
	36	37	0.02	37	0.02	37	18.48	37	0.21
26	23	31	0.15	31	0.20	31	132.67	31	184.10
	21	27	0.17	27	0.21	26	599.16	27	599.33
	32	32	0.02	32	0.01	32	44.51	32	13.16
	57	57	0.07	57	0.14	53	598.94	57	29.98
	25	31	0.21	31	0.30	25	599.00	30	599.06
50	43	54	0.68	54	1.46	44	599.04	46	598.91
	30	40	149.36	40	114.10	36	599.22	36	599.15
	37	49	0.51	49	0.40	43	599.18	46	599.09
	74	86	0.28	86	0.33	61	599.21	44	599.14
	83	84	0.13	84	0.26	66	599.29	83	599.13
80	82	83	0.16	83	0.20	25	599.17	82	599.17
	79	84	1.01	84	2.08	43	599.24	79	599.15
	86	86	0.12	86	0.55	59	599.43	86	5.25
	148	153	0.77	153	1.10	0	599.39	149	599.30
	148	150	1.99	150	2.34	0	599.46	0	599.32
140	152	153	2.27	153	3.36	0	599.47	0	599.39
	151	152	0.57	152	5.50	0	599.40	151	599.59
	139	153	0.36	153	0.72	0	599.43	139	599.28

# of	MIP/CP			MIP/CP_H			CP/CP			CP/CP_H		
Nodes	obj	runtime	itrs	obj	runtime	itrs	obj	runtime	itrs	obj	runtime	itrs
	5	0.00	1	5	0.01	1	5	0.00	1	5	0.00	1
	5	0.00	2	5	0.00	2	5	0.01	2	5	0.00	2
16	16	0.08	3	16	0.09	3	16	0.00	1	16	0.01	1
	21	0.00	1	21	0.00	1	21	0.01	2	20	0.00	1
	21	0.00	1	21	0.00	1	21	0.13	5	14	0.00	1

	5	0.01	3	5	0.01	3	5	0.02	5	5	0.01	3
	37	0.01	1	37	0.01	1	37	0.21	4	30	0.01	1
26	31	0.86	16	31	0.86	16	31	1.22	19	31	1.11	16
	27	0.20	4	27	0.19	4	27	1.07	16	26	0.00	1
	32	0.25	4	32	0.01	1	32	0.29	5	27	0.01	1
	57	1.66	16	57	1.66	16	0	600.04	1613	56	0.02	1
	31	0.45	9	31	0.46	9	31	3.15	56	31	1.13	13
50	54	6.26	63	54	6.20	63	54	29.09	153	43	1.36	10
	40	1.00	19	40	1.01	19	40	0.45	16	34	0.10	2
	49	3.81	46	49	3.82	46	49	75.40	176	49	23.37	172
	86	21.04	137	86	21.02	137	86	261.75	547	86	253.21	629
	84	3.39	31	84	3.39	31	84	337.47	645	84	467.04	762
80	83	1.53	14	83	1.53	14	83	295.67	809	83	65.31	170
	84	9.12	55	84	9.08	55	0	600.11	955	81	134.15	102
	86	8.55	54	86	0.02	1	86	124.96	251	86	0.05	1
	153	14.47	34	153	14.54	34	101	600.09	467	94	600.04	431
	150	9.91	44	150	9.96	44	0	601.51	517	76	610.48	17
140	153	16.22	71	153	16.23	71	45	600.05	301	153	562.07	75
	152	43.66	94	152	43.80	94	64	600.25	442	32	600.65	8
	148	600.37	1044	148	600.29	1042	40	601.74	3	49	600.08	347

Appendix C

Experimental Results for STPOB

Similarly structured as Appendix A, the tables in this section present the details of the experimental results on the generated dataset.

Big Time Windows									
# of	Heuristic	CP		CP_H		MIP		MIP_H	
Nodes	obj	obj	runtime	obj	runtime	obj	runtime	obj	runtime
	230.75	230.75	0.02	230.75	0.01	230.75	0.05	230.75	0.01
	264.98	264.98	0.01	264.98	0.02	264.98	0.11	264.98	0.01
16	241.19	241.19	0.01	241.19	0.01	241.19	1.88	241.19	0.01
	389.26	386.36	5.01	386.36	1.75	386.36	0.04	386.36	0.04
	219.36	219.36	0.00	219.36	0.01	219.36	0.21	219.36	0.01
	375.08	375.08	0.04	375.08	0.02	375.08	20.93	375.08	0.05
	330.21	330.21	0.02	330.21	0.02	330.21	19.89	330.21	0.08
26	368.78	368.78	0.01	368.78	0.01	368.78	25.51	368.78	0.05
	359.04	359.04	0.02	359.04	0.02	359.04	79.38	359.04	0.04
	298.19	298.19	0.03	298.19	0.01	298.19	22.43	298.19	0.05
	663.54	663.54	0.13	663.54	0.04	673.22	598.46	663.54	0.22
	827.73	777.85	600.03	791.12	600.04	873.56	598.77	810.39	598.76
50	637.24	614.60	6.56	614.60	5.03	640.43	598.65	637.24	598.97
	638.96	638.96	0.13	638.96	0.04	682.62	598.69	638.96	0.36
	587.67	580.98	0.08	580.98	0.26	607.45	595.45	580.98	89.97
	1020.66	1020.66	1.31	1020.66	0.46	1020.66	394.06	1020.66	1.63
	865.04	865.04	0.82	865.04	1.04	1199.84	597.32	865.04	1.03
80	1044.50	1044.50	0.93	1044.50	0.10	1089.00	597.80	1044.50	2.54
	1019.69	1019.69	0.95	1019.69	3.67	1019.69	576.36	1019.69	2.02
	1102.92	1102.92	1.72	1102.92	0.13	1642.69	598.22	1102.92	0.87
	1607.81	1607.81	7.02	1607.81	0.38	5359.35	598.87	1607.81	32.58
	1569.32	1569.32	11.46	1569.32	15.44	4319.69	598.91	1569.32	25.25
140	1621.31	1621.31	7.84	1621.31	2.56	5404.35	598.91	1621.31	27.01
	1641.86	1641.86	3.92	1641.86	6.45	5472.85	598.97	1641.86	14.90
	1574.66	1574.66	8.26	1574.66	14.99	5248.85	598.95	1574.66	28.52

# of	MIP/CP			MIP/CP_H			CP/CP			CP/CP_H		
Nodes	obj	runtime	itrs	obj	runtime	itrs	obj	runtime	itrs	obj	runtime	itrs
16	230.75	0.00	1	230.75	0.01	1	230.75	0.01	1	230.75	0.01	1
	264.98	0.00	1	264.98	0.01	1	264.98	0.01	1	264.98	0.01	1
	241.19	0.01	1	241.19	0.00	1	241.19	0.00	1	241.19	0.00	1
	386.36	0.01	1	386.36	0.00	1	386.36	0.01	1	386.36	0.01	1
	219.36	0.01	1	219.36	0.00	1	219.36	0.00	1	219.36	0.01	1
26	375.08	0.01	1	375.08	0.00	1	375.08	0.02	1	375.08	0.02	1
	330.21	0.01	1	330.21	0.00	1	330.21	0.02	1	330.21	0.02	1
	368.78	0.01	1	368.78	0.02	1	368.78	0.02	1	368.78	0.01	1
	359.04	0.01	1	359.04	0.00	1	359.04	0.90	1	359.04	0.90	1
	298.19	0.01	1	298.19	0.01	1	298.19	0.01	1	298.19	0.02	1
50	663.54	0.02	1	663.54	0.02	1	899.79	600.01	2	836.02	600.01	2
	1160.47	600.00	2	1160.47	600.00	2	777.85	471.11	7	777.85	461.71	7
	614.60	599.25	2	2630.89	598.33	2	614.60	123.91	1	614.60	123.61	1
	638.96	0.02	1	2473.26	0.04	1	689.11	600.02	2	689.11	600.00	2
	580.98	0.02	1	866.13	0.02	1	717.83	600.00	2	717.83	600.01	2
80	1020.66	0.03	1	1020.66	0.05	1	1360.37	600.05	3	1360.37	600.01	2
	865.04	0.02	1	865.04	0.04	1	960.59	600.01	2	960.59	600.05	3
	1044.50	0.02	1	1044.50	0.03	1	1316.94	600.02	1	1314.94	600.00	1
	1019.69	0.02	1	1019.69	0.03	1	1291.46	600.01	2	1291.46	600.00	2
	1102.92	0.03	1	1102.92	0.04	1	1237.64	600.01	2	1237.64	600.00	2
14	1607.81	0.06	1	1607.81	0.09	1	1607.81	0.27	1	1607.81	1.53	1
	1943.26	600.00	2	1569.32	0.10	1	3270.63	600.01	2	3270.63	600.00	2
	1621.31	0.05	1	1621.31	0.09	1	2988.09	600.01	2	2988.09	600.01	2
	3638.99	600.01	1	1641.86	0.12	1	3586.18	600.01	1	3586.18	600.01	1
	1574.66	0.14	1	1574.66	0.09	1	3185.29	600.00	2	3185.29	600.00	2

Normal Time Windows									
# of	Heuristic	CP		CP_H		MIP		MIP_H	
Nodes	obj	obj	runtime	obj	runtime	obj	runtime	obj	runtime
16	394.48	232.32	0.02	232.32	0.02	232.32	1.61	232.32	1.92
	277.38	247.59	0.01	247.59	0.01	247.59	2.35	247.59	1.07
	322.00	270.79	0.00	270.79	0.01	270.79	0.05	270.79	0.04
	243.62	216.48	0.03	216.48	0.02	216.48	0.97	216.48	2.26
	284.71	264.93	0.12	264.93	0.16	264.93	0.09	264.93	0.05
26	448.25	391.70	2.17	391.70	1.33	391.70	181.50	391.70	0.61
	434.95	332.03	0.05	332.03	0.05	339.05	598.97	338.91	599.14
	394.96	392.81	0.02	392.81	0.03	392.81	599.02	392.81	487.58
	438.34	438.34	12.38	438.34	6.98	438.34	599.12	438.34	599.23

	485.69	338.69	0.06	338.69	0.04	338.69	20.29	338.69	20.46
50	755.34	697.10	600.02	697.10	183.86	890.74	598.73	714.66	598.98
	719.97	589.46	2.54	589.46	0.78	644.75	599.02	630.17	598.89
	691.75	659.33	0.56	659.33	2.54	756.51	598.69	691.75	598.95
	786.80	770.15	0.06	770.15	1.02	905.25	598.83	770.15	545.82
	712.87	695.54	0.19	695.54	0.46	798.52	598.75	712.87	598.90
80	1252.10	1199.53	3.71	1199.53	2.24	1445.71	598.76	1236.81	599.03
	1135.06	1105.66	1.40	1105.66	4.15	1181.17	598.54	1110.56	598.95
	1059.62	1014.94	210.32	1014.94	600.08	1169.67	598.89	1047.69	599.09
	1372.29	1372.29	0.34	1372.29	1.02	1659.67	598.48	1372.29	599.02
	1088.54	1088.54	0.78	1088.54	1.97	1538.04	598.97	1088.54	86.61
140	1557.09	1557.09	20.34	1557.09	15.40	5190.30	598.73	1560.60	599.23
	1774.77	1774.77	8.01	1774.77	21.04	5256.50	599.01	1774.77	599.24
	1560.21	1560.21	12.25	1560.21	17.34	5200.70	598.88	5200.70	599.13
	1731.99	1731.99	7.68	1731.99	13.01	5290.40	598.87	1731.99	598.93
	1795.42	1795.42	5.96	1795.42	6.26	5463.80	599.06	1795.42	599.05

# of	MIP/CP			MIP/CP_H			CP/CP			CP/CP_H		
Nodes	obj	runtime	itrs	obj	runtime	itrs	obj	runtime	itrs	obj	runtime	itrs
16	232.32	0.01	1	232.32	0.01	1	232.32	0.01	1	232.32	0.01	1
	247.59	0.00	1	247.59	0.00	1	247.59	0.01	1	247.59	0.01	1
	270.79	0.00	1	270.79	0.01	1	270.79	0.01	1	270.79	0.00	1
	216.48	0.00	1	216.48	0.00	1	216.48	0.00	1	216.48	0.02	1
	264.93	0.00	1	264.93	0.01	1	264.93	0.01	1	264.93	0.01	1
26	391.70	0.01	1	391.70	0.00	1	391.70	0.02	1	391.70	0.02	1
	332.03	0.86	3	332.03	0.86	3	332.03	0.62	3	332.03	0.63	3
	392.81	0.01	1	392.81	0.01	1	392.81	0.02	1	392.81	0.02	1
	438.34	0.02	2	438.34	0.02	2	438.34	0.29	3	438.34	0.30	3
	338.69	0.05	1	338.69	0.04	1	338.69	0.54	3	338.69	0.54	3
50	697.10	24.92	60	697.10	24.82	60	697.10	195.82	63	697.10	221.53	62
	866.13	600.01	267	593.24	600.00	269	589.46	20.39	50	589.46	20.31	50
	659.33	5.30	2	659.33	5.31	2	659.33	5.24	4	659.33	5.22	4
	770.15	92.80	23	770.15	92.94	23	770.15	4.04	5	770.15	4.00	5
	695.54	154.34	12	695.54	153.47	12	929.79	600.00	2	914.78	600.01	3
80	1199.53	117.78	12	1199.53	117.88	12	1199.53	91.53	47	1199.53	62.43	47
	1105.66	256.60	9	1105.66	255.16	9	1430.85	600.00	2	1640.04	600.00	2
	1014.94	158.92	164	1014.94	159.11	164	1085.15	183.25	58	1085.15	181.91	58
	1372.29	0.07	2	1372.29	0.04	1	1893.55	600.00	2	1924.24	600.00	2
	1088.54	63.67	9	1088.54	0.04	1	1403.47	600.02	3	1403.47	600.00	2
140	2630.89	600.00	66	1212.91	600.01	67	1557.09	93.99	48	1557.09	152.38	54
	2732.20	600.00	4	1774.77	0.09	1	3323.24	600.00	1	3233.64	600.01	1
	2473.26	600.00	3	516.75	600.01	3	1560.21	397.84	22	1560.21	502.61	42
	3591.33	600.01	2	1731.99	0.08	1	1731.99	46.11	10	1731.99	47.01	10
	3730.85	600.02	3	1795.42	0.08	1	3651.08	604.38	2	3651.08	604.10	2

Small Time Windows									
# of	Heuristic	CP		CP_H		MIP		MIP_H	
Nodes	obj	obj	runtime	obj	runtime	obj	runtime	obj	runtime
16	590.42	590.42	0.02	590.42	0.00	590.42	0.02	590.42	0.01
	628.18	628.18	0.02	628.18	0.01	628.18	0.03	628.18	0.05
	414.68	352.04	0.01	352.04	0.01	352.04	0.65	352.04	0.74
	423.53	307.04	0.08	307.04	0.05	307.04	0.12	307.04	0.11
	527.37	527.37	0.01	527.37	0.01	527.37	0.04	527.37	0.02
26	935.94	935.94	0.01	935.94	0.01	935.94	9.96	935.94	10.25
	372.71	372.71	0.02	372.71	0.01	372.71	1.26	372.71	0.07
	544.82	403.77	0.06	403.77	0.27	403.77	73.72	403.77	46.50
	921.88	892.24	0.02	892.24	0.02	892.24	599.23	892.24	599.31
	457.58	444.33	0.05	444.33	0.08	444.33	140.00	444.33	257.44
50	746.37	722.47	0.67	722.47	2.46	733.42	598.74	742.77	598.90
	1660.31	1526.14	0.15	1526.14	0.19	1635.67	598.89	1526.14	598.90
	1123.59	964.30	0.26	964.30	0.18	1188.51	598.22	1044.07	599.03
	1519.23	1495.33	600.02	1495.33	458.56	1500.02	598.97	1500.02	598.98
	1456.39	1334.65	0.30	1334.65	0.52	1389.43	598.89	1369.06	599.04
80	1835.64	1517.46	2.88	1517.46	5.54	2594.16	598.84	1701.83	599.08
	1456.77	1442.40	0.23	1442.40	0.97	2643.18	599.10	1447.09	599.26
	2218.71	2218.71	0.09	2218.71	0.08	2579.54	599.21	2218.71	599.30
	1603.55	1508.79	3.12	1508.79	8.56	2749.17	598.95	1603.55	599.11
	1161.53	1128.07	15.11	1128.07	18.02	1780.87	598.99	1507.29	599.25
140	2271.50	2260.12	1.10	2260.12	2.65	4641.65	599.20	4641.65	599.31
	2433.79	2376.42	2.56	2376.42	3.80	4704.55	599.44	2433.50	599.32
	2228.10	2208.21	15.32	2208.21	8.87	5329.93	599.09	5329.93	599.30
	2625.65	2563.28	2.50	2563.28	2.47	5379.80	599.00	5379.80	599.04
	2494.99	2447.90	600.17	2447.90	421.15	4232.23	599.07	4232.23	599.36

# of	MIP/CP			MIP/CP_H			CP/CP			CP/CP_H		
Nodes	obj	runtime	itrs	obj	runtime	itrs	obj	runtime	itrs	obj	runtime	itrs
16	590.42	0.01	1	590.42	0.00	1	590.42	0.01	1	590.42	0.01	1
	628.18	0.01	2	628.18	0.00	2	628.18	0.01	2	628.18	600.05	15860
	352.04	0.00	1	352.04	0.00	1	352.04	0.01	1	352.04	0.02	1
	307.04	0.08	2	307.04	0.08	2	307.04	0.08	2	404.03	600.00	6585
	527.37	0.00	1	527.37	0.01	1	527.37	0.00	1	527.37	0.01	1
26	935.94	0.00	1	935.94	0.00	1	935.94	0.01	1	935.94	0.01	1
	372.71	0.01	1	372.71	0.01	1	372.71	0.01	1	372.71	0.02	1
	403.77	0.09	2	403.77	0.07	2	403.77	0.11	2	520.53	600.02	6478
	892.24	0.00	1	892.24	0.00	1	892.24	0.01	1	892.24	0.01	1

	444.33	0.11	3	444.33	0.11	1	542.73	0.01	1	542.73	600.00	6384
50	722.47	2.66	12	722.47	2.66	12	722.47	253.13	29	1044.01	600.00	9
	1526.14	0.86	11	1526.14	0.85	11	1526.14	1.22	11	1635.70	600.01	2089
	964.30	9.01	32	964.30	8.99	32	964.30	19.15	65	1250.64	600.00	372
	1495.33	0.30	3	1495.33	0.30	3	1495.33	0.01	1	1495.33	0.04	1
	1334.65	0.37	4	1334.65	0.36	4	1334.65	0.30	2	1376.30	600.01	2524
80	1517.46	80.08	386	1517.46	79.80	386	1539.00	600.21	1304	1539.00	600.09	1305
	1442.40	62.64	306	1442.40	62.79	306	1442.40	16.16	60	1442.40	16.08	60
	2218.71	0.04	2	2218.71	0.03	1	2218.71	6.26	20	2218.71	0.10	1
	1508.79	22.76	143	1508.79	22.65	143	1624.22	42.36	51	1624.22	42.68	51
	1128.07	22.78	117	1128.07	22.69	117	1128.07	69.45	86	1128.07	70.73	86
140	2260.12	16.25	75	2260.12	16.19	75	2260.12	85.07	281	2260.12	84.86	281
	2376.42	17.76	109	2376.42	17.76	109	2376.42	175.95	373	2376.42	175.12	373
	2208.21	8.22	33	2208.21	8.23	33	2208.21	162.36	237	2208.21	160.11	237
	2563.28	27.61	117	2563.28	27.53	117	2569.30	600.20	1039	2569.30	600.01	1050
	2447.90	121.83	298	2447.90	120.62	397	2635.04	600.18	138	3627.92	600.00	1493

Bibliography

[1] Claudia Archetti, Dominique Feillet, Alain Hertz, and Maria Grazia Speranza. The capacitated team orienteering and profitable tour problems. *Journal of the Operational Research Society*, 60(6):831–842, 2009.

[2] DG Baklagis, George Dikas, and Ioannis Minis. The team orienteering pick-up and delivery problem with time windows and its applications in fleet sizing. *RAIRO-Operations Research*, 50(3):503–517, 2016.

[3] Roberto Baldacci, Enrico Bartolini, and Aristide Mingozzi. An exact algorithm for the pickup and delivery problem with time windows. *Operations research*, 59(2):414–426, 2011.

[4] Roberto Baldacci, Nicos Christofides, and Aristide Mingozzi. An exact algorithm for the vehicle routing problem based on the set partitioning formulation with additional cuts. *Mathematical Programming*, 115(2):351–385, 2008.

[5] Michel L Balinski and Richard E Quandt. On an integer program for a delivery problem. *Operations Research*, 12(2):300–304, 1964.

[6] Martin J Beckmann. Decision and team problems in airline reservations. *Econometrica: Journal of the Econometric Society*, pages 134–145, 1958.

[7] Gerardo Berbeglia, Jean-François Cordeau, Irina Gribkovskaia, and Gilbert Laporte. Static pickup and delivery problems: a classification scheme and survey. *Top*, 15(1):1–31, 2007.

[8] Sylvain Boussier, Dominique Feillet, and Michel Gendreau. An exact algorithm for team orienteering problems. *4OR*, 5(3):211–230, 2007.

[9] Steven E Butt and David M Ryan. An optimal solution procedure for the multiple tour maximum collection problem using column generation. *Computers & Operations Research*, 26(4):427–441, 1999.

[10] Jose Caceres-Cruz, Pol Arias, Daniel Guimarans, Daniel Riera, and Angel A Juan. Rich vehicle routing problem: Survey. *ACM Computing Surveys (CSUR)*, 47(2):32, 2015.

[11] Richard E Chatwin. Multiperiod airline overbooking with a single fare class. *Operations Research*, 46(6):805–819, 1998.

[12] Claudio Contardo and Rafael Martinelli. A new exact algorithm for the multi-depot vehicle routing problem under capacity and route length constraints. *Discrete Optimization*, 12:129–146, 2014.

[13] Jean-François Cordeau. A branch-and-cut algorithm for the dial-a-ride problem. *Operations Research*, 54(3):573–586, 2006.

[14] Jean-François Cordeau, Michel Gendreau, Alain Hertz, Gilbert Laporte, and Jean-Sylvain Sormany. New heuristics for the vehicle routing problem. In *Logistics systems: design and optimization*, pages 279–297. Springer, 2005.

[15] Jean-François Cordeau and Gilbert Laporte. The dial-a-ride problem: models and algorithms. *Annals of operations research*, 153(1):29, 2007.

[16] Jean-François Cordeau, Gilbert Laporte, and Stefan Ropke. Recent models and algorithms for one-to-one pickup and delivery problems. *The vehicle routing problem: latest advances and new challenges*, 43:327–357, 2008.

[17] Jean-FrançoisF Cordeau, Guy Desaulniers, Jacques Desrosiers, Marius M Solomon, and François Soumis. *The VRP with time windows*. Montréal: Groupe d'études et de recherche en analyse des décisions, 2000.

[18] Guy Desaulniers, Oli BG Madsen, and Stefan Ropke. The vehicle routing problem with time windows. *Vehicle routing: Problems, methods, and applications*, 18:119–159, 2014.

[19] G. Dikas and Minis I. Scheduled Paratransit Transport Systems. *Transportation Research*, Part B(67):18–34, 2014.

[20] Yvan Dumas, Jacques Desrosiers, and Francois Soumis. The pickup and delivery problem with time windows. *European journal of operational research*, 54(1):7–22, 1991.

[21] Jeffrey C Ely, Daniel F Garrett, and Toomas Hinnosaar. Overbooking. *Journal of the European Economic Association*, page jvw025, 2017.

[22] Dominique Feillet, Pierre Dejax, and Michel Gendreau. Traveling salesman problems with profits. *Transportation Science*, 39(2):188–205, 2005.

[23] Matteo Fischetti, Juan Jose Salazar Gonzalez, and Paolo Toth. Solving the orienteering problem through branch-and-cut. *INFORMS Journal on Computing*, 10(2):133–148, 1998.

[24] Maria Gabriela S Furtado, Pedro Munari, and Reinaldo Morabito. Pickup and delivery problem with time windows: a new compact two-index formulation. *Operations Research Letters*, 45(4):334–341, 2017.

[25] Ridvan Gedik, Emre Kirac, Ashlea Bennet Milburn, and Chase Rainwater. A constraint programming approach for the team orienteering problem with time windows. *Computers & Industrial Engineering*, 107:178–195, 2017.

[26] Michel Gendreau, Gilbert Laporte, and Frederic Semet. A branch-and-cut algorithm for the undirected selective traveling salesman problem. *Networks*, 32(4):263–273, 1998.

[27] Michel Gendreau, Gilbert Laporte, and Frédéric Semet. A tabu search heuristic for the undirected selective travelling salesman problem. *European Journal of Operational Research*, 106(2-3):539–545, 1998.

[28] Suyun Geng et al. The complexity of the 0/1 multi-knapsack problem. *Journal of Computer Science and Technology*, 1(1):46–50, 1986.

[29] Billy E Gillett and Leland R Miller. A heuristic algorithm for the vehicle-dispatch problem. *Operations research*, 22(2):340–349, 1974.

[30] Inge Li Gørtz, Viswanath Nagarajan, and R Ravi. Minimum makespan multi-vehicle dial-a-ride. In *European Symposium on Algorithms*, pages 540–552. Springer, 2009.

[31] Aldy Gunawan, Hoong Chuin Lau, and Pieter Vansteenwegen. Orienteering problem: A survey of recent variants, solution approaches and applications. *European Journal of Operational Research*, 255(2):315–332, 2016.

[32] Stefan Heinz, Wen-Yang Ku, and J Christopher Beck. Recent improvements using constraint integer programming for resource allocation and scheduling. In *International Conference on AI and OR Techniques in Constriant Programming for Combinatorial Optimization Problems*, pages 12–27. Springer, 2013.

[33] Hipólito Hernández-Pérez and Juan-José Salazar-González. The multi-commodity one-to-one pickup-and-delivery traveling salesman problem. *European Journal of Operational Research*, 196(3):987–995, 2009.

[34] John N Hooker and Greger Ottosson. Logic-based benders decomposition. *Mathematical Programming*, 96(1):33–60, 2003.

[35] Stefan Irnich. A multi-depot pickup and delivery problem with a single hub and heterogeneous vehicles. *European Journal of Operational Research*, 122(2):310–328, 2000.

[36] Brian Kallehauge. Formulations and exact algorithms for the vehicle routing problem with time windows. *Computers & Operations Research*, 35(7):2307–2330, 2008.

[37] Brian Kallehauge, Jesper Larsen, Oli BG Madsen, and Marius M Solomon. *Vehicle routing problem with time windows*, pages 67–98. Springer, 2005.

[38] Taehyeong Kim and Ali Haghani. Model and algorithm considering time-varying travel times to solve static multidepot dial-a-ride problem. *Transportation Research Record: Journal of the Transportation Research Board*, pages 68–77, 2011.

[39] Linda R LaGanga and Stephen R Lawrence. Clinic overbooking to improve patient access and increase provider productivity. *Decision Sciences*, 38(2):251–276, 2007.

[40] Kin-Keung Lai and Wan-Lung Ng. A stochastic approach to hotel revenue optimization. *Computers & Operations Research*, 32(5):1059–1072, 2005.

[41] Gilbert Laporte. Fifty years of vehicle routing. *Transportation Science*, 43(4):408–416, 2009.

[42] Gilbert Laporte, Michel Gendreau, J-Y Potvin, and Frédéric Semet. Classical and modern heuristics for the vehicle routing problem. *International transactions in operational research*, 7(4-5):285–300, 2000.

[43] Gilbert Laporte and Silvano Martello. The selective travelling salesman problem. *Discrete applied mathematics*, 26(2-3):193–207, 1990.

[44] Hoong Chuin Lau and Zhe Liang. Pickup and delivery with time windows: Algorithms and test case generation. *International Journal on Artificial Intelligence Tools*, 11(03):455–472, 2002.

[45] Jan Karel Lenstra and AHG Kan. Complexity of vehicle routing and scheduling problems. *Networks*, 11(2):221–227, 1981.

[46] Yun-Chia Liang, Sadan Kulturel-Konak, and Alice E Smith. Meta heuristics for the orienteering problem. In *Evolutionary Computation, 2002. CEC'02. Proceedings of the 2002 Congress on*, volume 1, pages 384–389. IEEE, 2002.

[47] Quan Lu and Maged Dessouky. An exact algorithm for the multiple vehicle pickup and delivery problem. *Transportation Science*, 38(4):503–514, 2004.

[48] Emanuel Melachrinoudis, Ahmet B Ilhan, and Hokey Min. A dial-a-ride problem for client transportation in a health-care organization. *Computers & Operations Research*, 34(3):742–759, 2007.

[49] Roberto Montemanni and Luca Maria Gambardella. An ant colony system for team orienteering problems with time windows. *Foundation Of Computing And Decision Sciences*, 34(4):287, 2009.

[50] Kumar Muthuraman and Mark Lawley. A stochastic overbooking model for outpatient clinical scheduling with no-shows. *Iie Transactions*, 40(9):820–837, 2008.

[51] United Nations. World population ageing 2015. Technical report, Department of Economic and Social Affairs, Population Davision, 2015. (ST/ESA/SER.A/390).

[52] Sophie N Parragh, Karl F Doerner, and Richard F Hartl. A survey on pickup and delivery models part ii: Transportation between pickup and delivery locations. *Journal für Betriebswirtschaft*, 58(2):81–117, 2008.

[53] Sophie N Parragh, Karl F Doerner, and Richard F Hartl. A survey on pickup and delivery problems: Part ii: Transportation between pickup and delivery locations. *Journal für Betriebswirtschaft*, 58:81–117, 2008.

[54] Xiaoqiu Qiu, Stefan Feuerriegel, and Dirk Neumann. Making the most of fleets: A profit-maximizing multi-vehicle pickup and delivery selection problem. *European Journal of Operational Research*, 259(1):155–168, 2017.

[55] Ted K Ralphs, Leonid Kopman, William R Pulleyblank, and Leslie E Trotter. On the capacitated vehicle routing problem. *Mathematical programming*, 94(2):343–359, 2003.

[56] Philippe Refalo. Impact-based search strategies for constraint programming. *CP*, 3258:557–571, 2004.

[57] Stefan Ropke and Jean-François Cordeau. Branch and cut and price for the pickup and delivery problem with time windows. *Transportation Science*, 43(3):267–286, 2009.

[58] Vahid Roshanaei, Curtiss Luong, Dionne M Aleman, and David Urbach. Propagating logic-based benders decomposition approaches for distributed operating room scheduling. *European Journal of Operational Research*, 257(2):439–455, 2017.

[59] Michele Samorani and Linda LaGanga. A stochastic programming approach to improve overbooking in clinic appointment scheduling. In *POMS annual meeting proceedings*, 2011.

[60] Mikkel Sigurd, David Pisinger, and Michael Sig. Scheduling transportation of live animals to avoid the spread of diseases. *Transportation Science*, 38(2):197–209, 2004.

[61] M Faith Tasgetiren. A genetic algorithm with an adaptive penalty function for the orienteering problem. *Journal of Economic & Social Research*, 4(2), 2002.

[62] Paolo Toth and Daniele Vigo. *Fast local search algorithms for the handicapped persons transportation problem*, pages 677–690. Springer, 1996.

[63] Paolo Toth and Daniele Vigo. *Vehicle routing: problems, methods, and applications*. SIAM, 2014.

[64] T. T. Tran, A. Araujo, and J. C. Beck. Decomposition methods for the parallel machine scheduling problem with setups. *INFORMS Journal on Computing*, 28(1):83–95, 2016.

[65] Pieter Vansteenwegen, Wouter Souffriau, Greet Vanden Berghe, and Dirk Van Oudheusden. Iterated local search for the team orienteering problem with time windows. *Computers & Operations Research*, 36(12):3281–3290, 2009.

[66] Pieter Vansteenwegen, Wouter Souffriau, and Dirk Van Oudheusden. The orienteering problem: A survey. *European Journal of Operational Research*, 209(1):1–10, 2011.

[67] Petr Vilím, Philippe Laborie, and Paul Shaw. Failure-directed search for constraint-based scheduling. In *International Conference on AI and OR Techniques in Constriant Programming for Combinatorial Optimization Problems*, pages 437–453. Springer, 2015.