

# Modelling and Solving the Senior Transportation Problem

Chang Liu, Dionne M. Aleman, and J. Christopher Beck

Department of Mechanical & Industrial Engineering  
University of Toronto, Toronto, Ontario M5S 3G8, Canada  
[{cliu,aleman,jcb}@mie.utoronto.ca](mailto:{cliu,aleman,jcb}@mie.utoronto.ca)

**Abstract.** This paper defines a novel transportation problem, the Senior Transportation Problem (STP), which is inspired by the elderly door-to-door transportation services provided by non-profit organizations. By extending the vehicle routing literature, we developed solution approaches including mixed integer programming (MIP), constraint programming (CP), two logic-based Benders decompositions (LBBD), and a construction heuristic. Empirical analyses on both randomly generated datasets and large real-life datasets are performed. CP achieved the best results, solving to optimality 89% of our real-life instances of up to 270 vehicles with 385 requests in under 600 seconds. The best LBBD model can only solve 17% of those instances to optimality. Further investigation of this somewhat surprising result indicates that, compared to the LBBD approaches, the pure CP model is able to find better solutions faster and then is able to use the bounds from these sub-optimal solutions to reduce the search space slightly more effectively than the decomposition models.

## 1 Introduction

As the world population ages, there is an increasing demand for transit options for elderly people who have difficulties accessing the regular public transit system but yet do not have disabilities that qualify them for specialized transit services. As a consequence, there are non-profit organizations that provide such “senior transportation” services in many communities. However, the resources for these services are often limited and many elders are put on a waiting list. Furthermore, due to lack of expertise and decision support tools, the schedules assigned to the drivers are often sub-optimal as many vehicles do not operate at full capacity. Therefore, finding optimal schedules is crucial for organizations to meet increasing demands.

The Senior Transportation Problem (STP) is a static 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. Due to the limited resources, not all requests can be met within the given time and, therefore, the problem is to select a subset of requests such that the total weight of all served requests is maximized. As some of the drivers operate on

a volunteer basis, the problem includes characteristics such as multiple depots, heterogeneous vehicles, and time windows on both locations and vehicles.

Our primary contributions are to formally define the STP and to provide solution techniques for the STP. Four exact methods based on mixed integer programming (MIP), constraint programming (CP), and two logic-based Benders decomposition (LBBD) models plus a construction heuristic are developed. We define and present detailed experimental results and analysis for each approach. On real-world data from a non-profit organization, CP performs substantially better than the other approaches, solving over 89% of the problems.

## 2 Background

In this section, we formally define the STP and review related work.

### 2.1 Problem Definition

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. Each vertex  $i \in \mathcal{V}$  is associated with a time window  $[E_i, L_i]$  and a service duration  $S_i$  corresponding to the time to be spent at location  $i$ . Each arc  $(i, j) \in \mathcal{A}$  has a non-negative routing time  $T_{i,j}$  satisfying the triangular inequality.

Let  $\mathcal{K} = \{1, \dots, |\mathcal{K}|\}$  represent the set of vehicles. Each vehicle  $k \in \mathcal{K}$  is associated with a starting and ending depot  $i_{k+}, i_{k-} \in \mathcal{D}$  where the vehicle must start and end, respectively. Multiple vehicles can share the depots but relocation of vehicles between depots is not allowed. Each vehicle also specifies its availability via time windows:  $[E_{i_{k+}}, L_{i_{k+}}]$  and  $[E_{i_{k-}}, L_{i_{k-}}]$ . If the vehicle is used, it must leave the starting depot during the first interval, perform all pickup and delivery requests assigned to it, and return to the ending depot during the second interval. Furthermore, vehicles differ in capacity with each vehicle  $k \in \mathcal{K}$  is associated with a maximum capacity  $C_k$ .

Let  $\mathcal{R} = \{1, \dots, |\mathcal{R}|\}$  represent the set of requests. Each request  $r$  is paired with a positive weight,  $W_r$ , denoting its importance. The total weight of served requests is the basis of the objective function. A request  $r \in \mathcal{R}$  has an associated pickup location  $i^+ \in \mathcal{N}$  and a delivery location  $i^- \in \mathcal{N}$ . In addition, each clients is restricted to a maximum ride time,  $F$ , on any vehicle. The time horizon is denoted by  $Z$ . The load size is positive for a pickup location vertex and negative for a delivery vertex,  $Q_i = -Q_{|\mathcal{R}|+i}, \forall i \in \mathcal{R}^+$ .

A *route* for vehicle  $k$  is a sequence of vertices,  $[i_{k+}, \dots, i_{k-}]$  and 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 sub-tours are allowed in any route;
7. The ride time of a client cannot exceed the maximum ride time  $F$ ;
8. All pickups and deliveries must be served within their time windows.

## 2.2 Related Work

There are three levels of decisions in the STP: the selection of requests, the assignment of vehicles to requests, and the routing of vehicles. Each decision problem is a well-studied problem on its own.

The selectivity and routing aspects of STP can be viewed as a Team Orienteering Problem (TOP) [8]. Alternatively, the routing and assignment of requests can be seen as a Pickup and Delivery Problem with Time Windows (PDPTW) [3,4] or a Dial-a-Ride Problem (DARP) [2]. In addition to minimizing total travel cost in the classical DARP, the Cordeau et al. noted that there can be other objectives of maximizing satisfied demand or quality of service but do not provide any formulation or reference. The PDPTW has been solved to optimality for loosely constrained instances of sizes up to 100 requests [7] while the DARP has only been solved to optimality for problems with 24 requests [2]. The most common solution approaches are heuristic.

The combination of the three decisions has only been looked at by two groups. Baklagis et al. [1] proposed a branch-and-price framework to tackle this problem and Qiu et al. [5] investigated a graph search and a maximum set packing formulation specially tailored for homogeneous fleets. These works however are missing three components that are critical to the STP: multiple depots, maximum ride time for a client, and heterogeneous fleets.

## 3 Models for the Senior Transportation Problem

We present four exact methods (MIP, CP, and two LBBD approaches) and one heuristic to solve the STP. Both LBBD approaches employ a CP sub-problem while they use MIP and CP for the master problem, respectively.

### 3.1 Mixed Integer Programming

In Figure 1, we present a MIP formulation adapted from the PDPTW formulation of Ropke & Cordeau [7]. The formulation uses three variables: a binary variable  $x_{k,i,j}$  and two continuous variables  $u_{k,i}$  and  $v_{k,i}$ .  $x_{k,i,j} = 1$  if vehicle  $k$  visits location  $j$  immediately after visiting location  $i$  and 0 otherwise.  $u_{k,i}$  indicates the time when vehicle  $k$  leaves location  $i \in \mathcal{V}$ . It is non-negative and less than or equal to the maximum time horizon  $Z$ . Variables  $v_{k,i}$  indicate the load of vehicle  $k$  after visiting location  $i \in \mathcal{V}$ . It is non-negative and is less than or equal to the vehicle capacity  $C_k$ .

The objective function (1) maximizes the sum of weights of served requests. Constraints (2) and (3) ensure that each vehicle leaves from its starting depot

$$\begin{aligned}
& \max \sum_{k \in \mathcal{K}} \sum_{r \in \mathcal{R}} \sum_{j \in \mathcal{V}} \left( W_r \times x_{k,i_{r+},j} \right) && (1) \\
\text{s.t. } & \sum_{j \in \mathcal{N}^+} x_{k,i_{k+},j} + x_{k,i_{k+},i_{k-}} = 1 && \forall k \in \mathcal{K} && (2) \\
& \sum_{i \in \mathcal{N}^-} x_{k,i,j_{k-}} + x_{k,i_{k+},i_{k-}} = 1 && \forall k \in \mathcal{K} && (3) \\
& \sum_{k \in \mathcal{K}} \sum_{j \in \mathcal{V}} x_{k,i_{r+},j} \leq 1 && \forall r \in \mathcal{R} && (4) \\
& \sum_{j \in \mathcal{V}} (x_{k,i,j} - x_{k,j,i}) = 0 && \forall k \in \mathcal{K}, i \in \mathcal{N} && (5) \\
& \sum_{j \in \mathcal{V}} (x_{k,i_{r+},j} - x_{k,j,i_{r-}}) = 0 && \forall k \in \mathcal{K}, r \in \mathcal{R} && (6) \\
& u_{k,j} \geq (u_{k,i} + T_{i,j} + S_j) - M \times (1 - x_{k,i,j}) && \forall k \in \mathcal{K}, i, j \in \mathcal{V} && (7) \\
& u_{k,i} \geq E_i - M \times \left( 1 - \sum_{j \in \mathcal{V}} x_{k,i,j} \right) && \forall k \in \mathcal{K}, i \in \mathcal{V} && (8) \\
& u_{k,i} \leq L_i - S_i + M \times \left( 1 - \sum_{j \in \mathcal{V}} x_{k,i,j} \right) && \forall k \in \mathcal{K}, i \in \mathcal{V} && (9) \\
& u_{k,i_{r+}} \leq u_{k,i_{r-}} && \forall k \in \mathcal{K}, r \in \mathcal{R} && (10) \\
& (u_{k,i_{r-}} - u_{k,i_{r+}}) \leq F && \forall k \in \mathcal{K}, r \in \mathcal{R} && (11) \\
& v_{k,j} \geq (v_{k,i} + Q_i) - M \times (1 - x_{k,i,j}) && \forall k \in \mathcal{K}, i, j \in \mathcal{V} && (12) \\
& x_{k,i,j} \in \{0, 1\} && \forall k \in \mathcal{K}, (i, j) \in \mathcal{A} && (13) \\
& 0 \leq u_{k,i} \leq Z && \forall k \in \mathcal{K}, i \in \mathcal{V} && (14) \\
& 0 \leq v_{k,i} \leq C_k && \forall k \in \mathcal{K}, i \in \mathcal{V} && (15)
\end{aligned}$$

Fig. 1: MIP model for the Senior Transportation Problem.

and ends at its ending depot. Constraint (4) allows for the selectivity of requests. Constant flow is enforced with Constraint (5). Constraint (6) specifies that the pickup and delivery locations of a request must be visited by the same vehicle. In Constraint (7), the travel time and service time of visited nodes are enforced. Constraints (8) and (9) make sure that each node that is visited must be visited within its time window. Constraint (10) imposes that pickup nodes must precede delivery nodes. Constraint (11) enforces that each ride does not exceed the maximum ride time. Constraint (12) keeps track of the load of each vehicle after visiting the node.

$$\max \sum_{r \in \mathcal{R}} (W_r \times \text{PresenceOf}(x_r)) \quad (16)$$

$$\text{s.t. } \text{Alternative}(x_i, X_i) \quad \forall i \in \mathcal{N} \quad (17)$$

$$\text{Before}(\bar{X}_{k,i_{r+}}, \bar{X}_{k,i_{r-}}) \quad \forall k \in \mathcal{K}, \forall r \in \mathcal{R} \quad (18)$$

$$\text{PresenceOf}(\bar{X}_{k,i_{r+}}) = \text{PresenceOf}(\bar{X}_{k,i_{r-}}) \quad \forall k \in \mathcal{K}, \forall r \in \mathcal{R} \quad (19)$$

$$\text{GetStartMax}(x_{i_{r-}}) - \text{GetEndMax}(x_{i_{r+}}) \leq F \quad \forall r \in \mathcal{R} \quad (20)$$

$$v_{k,i} = \text{StepAtStart}(\bar{X}_{k,i}, Q_i) \quad \forall k \in \mathcal{K}, \forall i \in \mathcal{N} \quad (21)$$

$$\sum_{i \in \mathcal{N}} v_{k,i} \leq C_k \quad \forall k \in \mathcal{K} \quad (22)$$

$$\text{First}(u_k, x_{i_{k+}}) \quad \forall k \in \mathcal{K} \quad (23)$$

$$\text{Last}(u_k, x_{i_{k-}}) \quad \forall k \in \mathcal{K} \quad (24)$$

$$\text{NoOverlap}(u_k) \quad \forall k \in \mathcal{K} \quad (25)$$

Fig. 2: CP model for the Senior Transportation Problem.

### 3.2 Constraint Programming

The CP formulation shown in Figure 2 employs interval variables linked using cumulative functions and sequence expressions. Each location  $i \in \mathcal{N}$  is an optional interval variable  $x_i$  that is bounded by its time windows and the length of its service time. We assume that each vehicle visits its depot locations regardless whether 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  $\bar{X}_{k,i}$  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  $u_k$  whose value is a permutation of locations visited by vehicle  $k$ .

The objective function (16) maximizes the total weight of fulfilled requests. Constraint (17) ensures that if location  $i$  is visited, then only one vehicle visits it. The sequence of pickup and delivery locations is ensured by Constraint (18). Constraint (19) enforces that if the pickup location 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 Constraint (20). Constraint (21) uses the cumulative function to keep track of the load of the vehicles after visiting location  $i$ . Constraint (22) enforces that the capacity of vehicle must not be exceeded. Constraints (23) and (24) indicate that each route must start at its associated start depot and end at its associated ending depot. The CP model uses the `NoOverlap` global constraint (25) to prevent sub-tours on each route.

$$\max \sum_{r \in \mathcal{R}} \sum_{k \in \mathcal{K}} (W_r \times \varphi_{k,r}) \quad (26)$$

$$\text{s.t. } \sum_{k \in \mathcal{K}} y_{k,i} \leq 1 \quad \forall i \in \mathcal{N} \quad (27)$$

$$\zeta_r = S_{i_{r+}} + T_{i_{r+}, j_{r-}} + S_{i_{r-}} \quad \forall r \in \mathcal{R} \quad (28)$$

$$Q_r \times \varphi_{k,r} \leq P_k \quad \forall k \in \mathcal{K}, r \in \mathcal{R} \quad (29)$$

$$(E_{i_{r+}} + \zeta_r) \times \varphi_{k,r} \leq L_{i_{k-}} \quad \forall k \in \mathcal{K}, r \in \mathcal{R} \quad (30)$$

$$(E_{i_{k+}} + \zeta_r) \times \varphi_{k,r} \leq L_{i_{r-}} \quad \forall k \in \mathcal{K}, r \in \mathcal{R} \quad (31)$$

$$\sum_{i \in \mathcal{N}} (y_{k,i} \times \mathcal{T}_i + S_i) + \mathcal{T}_{i_{k+}} + S_{i_{k+}} \leq L_{i_{k-}} - E_{i_{k+}} \quad \forall k \in \mathcal{K} \quad (32)$$

$$y_{k,r} = y_{k,r+|\mathcal{R}|} = \varphi_{k,r} \quad \forall k \in \mathcal{K}, r \in \mathcal{R} \quad (33)$$

$$y_{k,i}, \varphi_{k,r} \in \{0, 1\} \quad \forall k \in \mathcal{K}, i \in \mathcal{N}, r \in \mathcal{R} \quad (34)$$

*Benders Cuts*

Fig. 3: A MIP model for the LBBD Master Problem of the STP.

### 3.3 Logic-based Benders Decompositions

For the LBBD approaches, we divide the STP into a relaxed master problem and a number of sub-problems. The master problem finds the optimal relaxed assignment of requests to vehicles. Each sub-problem is, then, an optimization problem to find the optimal route given the assigned requests. If the optimal route for each sub-problem satisfies all requests assigned to it, then the global optimal solution has been found, otherwise, a Benders cut is produced. The LBBD models find a feasible global solution at every iteration since the route found in each sub-problem is feasible when the master objective value is ignored. We present one MIP and one CP formulation of the master problem and a single CP model for the sub-problem.

**MIP Master Problem** The master problem assigns each request into a vehicle using integer decision variables  $\varphi_{k,r}$  which equal 1 if request  $r$  is assigned to vehicle  $k$  and 0 otherwise, and  $y_{k,i}$  which equal 1 if location  $i$  is visited by vehicle  $k$  and 0 otherwise. Instead of modelling the exact travel distance between consecutive locations, we compute the minimum travel time from each location  $i$  to any other, denoted with  $\mathcal{T}_i$ . The sum of minimum travel time of all locations assigned to a vehicle must be less than or equal to the time availability of the vehicle. All other routing constraints are ignored in the master problem.

Figure 3 presents the model. The objective function (26) maximizes the total weight of all the requests served. Constraint (27) ensures all locations are visited at most once. The approximate length of a request,  $\zeta_r$  is modelled in Constraint

$$\begin{aligned}
& \max \text{ Objective (16)} \\
& \text{s.t. Constraints (17), (19)} \\
& \quad \text{EndBeforeStart}(x_{i_k+}, X_{i,k}) \quad \forall k \in \mathcal{K}, i \in \mathcal{N} \quad (35) \\
& \quad \text{EndBeforeStart}(X_{i,k}, x_{i_k-}) \quad \forall k \in \mathcal{K}, i \in \mathcal{N} \quad (36) \\
& \quad \text{EndBeforeStart}(x_{i_k+}, x_{i_k-}) \quad \forall k \in \mathcal{K} \quad (37) \\
& \quad \sum_{i \in \mathcal{N}} (\text{PresenceOf}(X_{i,k}) \times \mathcal{T}_i + S_i) \\
& \quad + \mathcal{T}_{i_k+} + S_{i_k+} \leq L_{i_k-} - E_{i_k+} \quad \forall k \in \mathcal{K} \quad (38) \\
& \quad \text{Benders Cuts}
\end{aligned}$$

Fig. 4: A CP model for the LBBD Master Problem of the STP.

(28) and Constraints (29)-(31) remove all infeasible requests from a specific vehicle. The relaxed total travel time for a vehicle is restricted to the time availability of the vehicle through Constraint (32). The relationship between the  $y_{k,i}$  and  $\varphi_{k,r}$  variables is established in Constraint (33) which also specifies that a corresponding pickup and delivery must be served by the same vehicle.

**CP Master Problem** The CP formulation presented in Figure 4 uses significantly fewer of variables than the full STP model. 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 3.2.

The objective and a number of constraints remain the same as in the full STP model (Figure 2). Since sequences are relaxed, no sequence variables are modelled but Constraints (35)-(37) ensure that each vehicle visits its starting depot and ending depot first and last, respectively. Finally, the distance relaxation is the same as in the MIP master problem represented by Constraint (38).

**CP Sub-problem** After the master problem assigns the requests, a sub-problem is created for each vehicle with at least two requests assigned. Each sub-problem is a single vehicle STP maximizing the total weight of served requests of those assigned by the master problem. If the sub-problem is able to schedule all the requests given to it, then the vehicle has a feasible assignment. Otherwise, the requests assigned to the vehicle are not feasible and the solution of the sub-problem is the optimal assignment for any proper subset of the assigned requests. The objective value of the sub-problem is then returned to the master problem as a Benders cut. With optimization sub-problems, at each iteration of the LBBD, the algorithm finds a globally feasible solution.

Let  $k^*$  represent the vehicle and  $\mathcal{R}^*$  the subset of requests assigned to  $k^*$  by the master problem. The CP formulation of the sub-problem uses three decision

$$\max \sum_{r \in \mathcal{R}^*} (W_r \times \text{PresenceOf}(x_{i_{r+}})) \quad (39)$$

s.t.

$$\text{Before}(u, x_{i+}, x_{i-}) \quad \forall i \in \mathcal{R}^* \quad (40)$$

$$\text{GetStartMax}(x_{i-}) - \text{GetEndMax}(x_{i+}) \leq F \quad \forall i \in \mathcal{R}^* \quad (41)$$

$$v_i = \text{StepAtStart}(v_i, Q_i) \quad \forall i \in \mathcal{N}^* \quad (42)$$

$$0 \leq \sum_{i \in \mathcal{N}^*} v_i \leq C_{k^*} \quad (43)$$

$$\text{First}(u, x_{i_{k^*+}}) \quad (44)$$

$$\text{Last}(u, x_{i_{k^*-}}) \quad (45)$$

$$\text{NoOverlap}(u) \quad (46)$$

Fig. 5: A CP model for the LBBD Sub-problem of the STP.

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.

Figure 5 presents the CP model for the subproblems. The objective function (39) maximizes the sum of weights of served requests. Constraint (40) makes sure that the pickup location is visited before the delivery location. The maximum ride time is enforced through Constraint (41). Constraints (42) and (43) keep track of the load of the vehicle after visiting each location and make sure that the load does not exceed the capacity of the vehicle at any location. Constraints (44) and (45) force the start and end of the sequence to be at the starting and ending depot, respectively. Finally, Constraint (46) takes into account the travel distances between locations for the sequence and eliminates sub-tours.

**Benders Cut** If a sub-problem schedules all the requests assigned to it, then it is feasible. Otherwise, an optimality cut is returned to the master problem. The cut specifies that given the subset of requests  $\mathcal{R}^*$  to vehicle  $k^*$ , the objective value of this combination cannot be larger than the sub-problem's optimal value denoted by  $z^*$ . This cut is modelled in a MIP formulation as in Inequality (47) and in a CP formulation as in Inequality (48).

$$\sum_{r \in \mathcal{J}_{h,k}} (\varphi_{k,r} \times W_r) \leq z^* \quad \forall k \in \mathcal{K}, h \in \{1, \dots, H-1\} \quad (47)$$

$$\sum_{r \in \mathcal{J}_{h,k}} (\text{PresenceOf}(X_{i_{r+},k} \times W_r) \leq z^* \quad \forall k \in \mathcal{K}, h \in \{1, \dots, H-1\} \quad (48)$$

---

**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

```
1 Sort  $\mathcal{R}$  based on descending order of  $\frac{\text{weight}}{\text{length}}$ ;  
2 Sort  $\mathcal{K}$  based on ascending order time window size;  
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  $v_2$ ;  
8       set start and end locations and start and end times for  $v_1$  and  $v_2$ ;  
9       insert  $v_1$  and  $v_2$  into  $\mathcal{K}$  based on the new time window lengths;  
10      break;  
11    end  
12  end  
13 end  
14 Regroup all pieces of the same vehicle to make scheduled routes;
```

---

### 3.4 A Construction Heuristic

We designed a simple heuristic to provide a feasible solution to the STP. It is used both as a basis of comparison with the exact techniques and as a warm-start solution.

Since the objective function is to maximize the total weight of the requests served, it is logical 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.

## 4 Experimental Results

In this section, we discuss the datasets used to test our approaches and present the performance of the five approaches proposed above, including using the heuristic to provide a starting solution for the exact techniques.

All 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.

### 4.1 Datasets

We generated 75 random datasets and extracted 280 problem instances from real world data provided by a partnering organization. In the generated problem

Table 1: Bounds on problem characteristics for generated datasets.

Characteristic		Lower Bound	Upper Bound
<b>Vehicle</b>			
number of vehicles		2	20
capacity		2	6
start depot and end depot service time		2	16
<b>Request</b>			
number of requests		6	50
size		1	3
weight		1	5
pickup location and delivery location service time		2	16
travel time		1	60
<b>Time Windows</b>			
small		80	180
normal		180	360
big		600	900

instances, we varied the number of requests and vehicles, and the sizes of the time window (TW) of each request and vehicle. We also experimented with three different time window sizes: big, normal and small. All other characteristics are generated randomly following a normal distribution. Table 1 outlines the lower and upper bounds of each characteristic.

From the historical records of our partnering organization, we extracted 72,883 requests and 54,494 vehicle records over 280 operating days from January 2015 to January 2016. A total of 280 datasets were created. On average, there are 260 requests per day and the maximum number of requests per day is 554. There are on average 187 vehicles available each day.

## 4.2 Results

Table 2 summarizes the results of all approaches on the generated datasets. CP solved all 75 (100%) instances to optimality in an average of 1.02 seconds, MIP/CP LBBD solved 71 (95%) instances with an average runtime of 21.78 seconds, CP/CP LBBD solved 49 (65%) instances with an average runtime of 110.14 seconds, and MIP solved 35 (48%) instances with an average of 90.00 seconds. The heuristic was able to find, but of course not prove, the optimal solution for 45 (60%) of the instances. In terms of relative solution quality compared to the optimal solutions, CP is again the best performer with the heuristic finding, on average, better solutions than both the MIP and CP/CP LBBD models.

Each of the four exact methods were then run with the heuristic solution as a warm start. Both MIP and CP/CP LBBD have a substantial improvement with the heuristic start. However, the only additional instances that they were able to solve to optimality were those for which the heuristic found an optimal solution. The runtime for MIP and CP/CP LBBD also improved significantly.

Table 2: Number of instances solved to optimality, average runtime, and average optimality gap for generated datasets. The “\*” indicates that while the heuristic found optimal solutions for 45 instances, it is, of course, not able to prove it.

Approach	# Instances Solved to Optimality	% Solved to Optimality	Average Runtime	Average Optimality Gap
Heuristic	45*	60.00%	<b>0.01</b>	4.13%
MIP	35	46.67%	90.00	30.68%
MIP_H	52	69.33%	22.36	1.80%
CP	<b>75</b>	100.00%	1.02	<b>0.00%</b>
CP_H	<b>75</b>	100.00%	2.38	<b>0.00%</b>
MIP/CP LBBB	71	94.67%	21.78	0.15%
MIP/CP LBBB_H	73	97.33%	2.54	0.09%
CP/CP LBBB	49	65.33%	110.14	18.95%
CP/CP LBBB_H	61	81.33%	42.58	10.85%

Table 3: Number of instances solved to optimality, average runtime, and average optimality gap for real world datasets.

Approach	# Instances Solved to Optimality	% Solved to Optimality	Average Runtime	Average Optimality Gap
CP	250	89.29%	126.74	3.03%
MIP/CP LBBB	47	16.79%	331.31	18.38%

The heuristic start only improves MIP/CP LBBB a little while it has very minimal effects on CP.

CP/CP LBBB exhibits lower solution quality than the heuristic, even when warm-started. Recall that the relaxed master problem often has better (but globally infeasible) solutions and so the warm start solution is replaced by a better master problem incumbent before the sub-problems are solved.

Given the performance of CP and MIP/CP LBBB, we apply them to the real world datasets. As shown in Table 3, out of 280 instances, 250 instances are solved to optimality with an average of 126.74 seconds using the pure CP model while the MIP/CP LBBB could only solve 47 instances in 331.31 seconds.

The evolution of runtime of the CP model as the problem sizes of the real instances increase is shown in Figure 6. It can be observed that there is an approximately linear increase in runtime up to about 400 nodes (with some outliers) but with larger problems, the runtime substantially increases.

We also ran CP with an 8-hour time limit. An additional 21 instances were solved to optimality but nine instances are still open. Thus Table 4 reports the solution quality relative to the best known solution for the real world datasets.

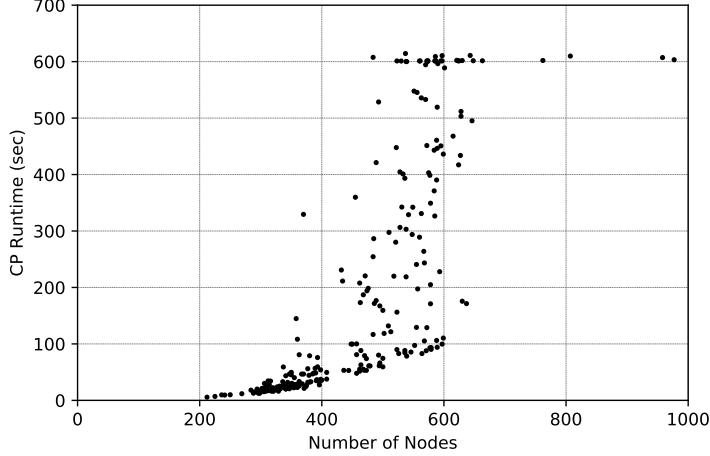


Fig. 6: CP Runtime of CHATS instances over number of vertices.

Table 4: 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%	<b>18.38%</b>

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. 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 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.

## 5 Analyses

The strong results for CP compared to the LBBD approaches differs from much of the literature. We have two hypotheses for why the CP model is working well. First, the CP model finds better feasible solutions faster. Second, these solutions result in more back-propagation from the lower-bound on the objective function, creating greater impact of search space reduction [6] compared to the LBBD models. In this section, we explore both of these ideas.

### 5.1 CP and LBBD First Solutions

We run the CP model and both LBBD approaches until finding a first feasible solution, recording the objective value and the time to find the solution. The

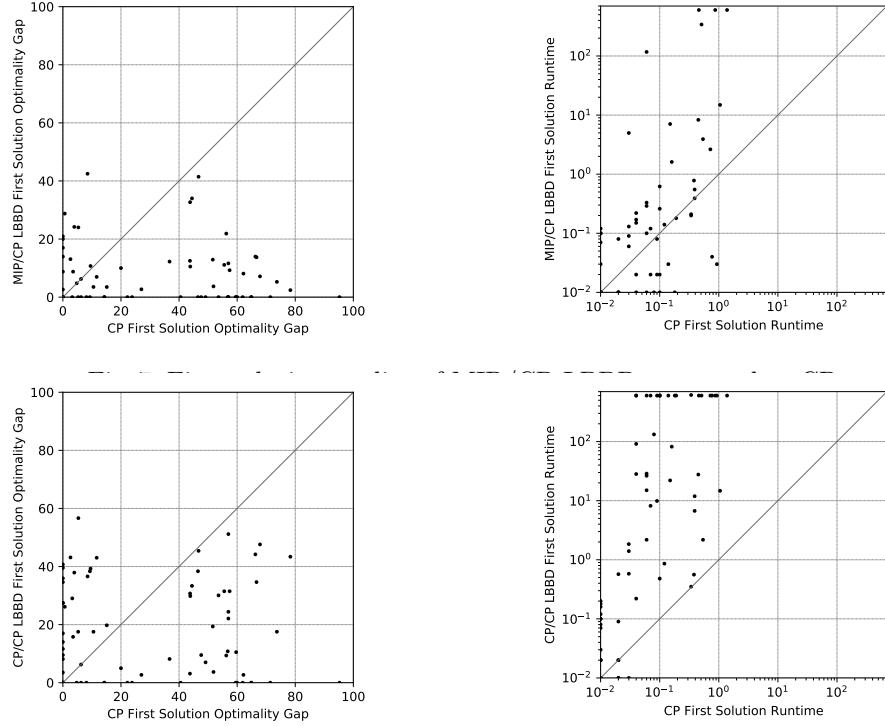


Fig. 8: First solution quality of MIP/CP LBBD compared to CP.

objective value,  $z'$ , is compared to the known optimal solution,  $z$ . The first solution gap is computed as  $(z - z')/z$ . Figures 7 and 8, respectively, compare the MIP/CP LBBD approach and the CP/CP LBBD approach to the CP model.

For the LBBD approaches, the first feasible solution is often the actual optimal solution. Therefore, while the first solution optimality gap of the LBBD approaches is usually better than the CP model, the time to find these solutions for the LBBD approaches is much longer.

To measure the effect of the first solution on the different approaches, we used 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 but 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 9.

For big time windows, some instances are solved more quickly with the warm start solution. However, on average, the run-times with or without the warm-start are the same. As with the CP/CP LBBD\_H results in Table 2, in many cases, the warm start solution provided by the CP model is not as good as the first master problem solution and thus is discarded.

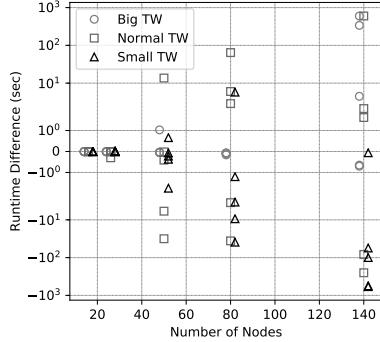


Fig. 9: Runtime difference of pure MIP/CP LBBB minus MIP/CP LBBB with CP starting solution.

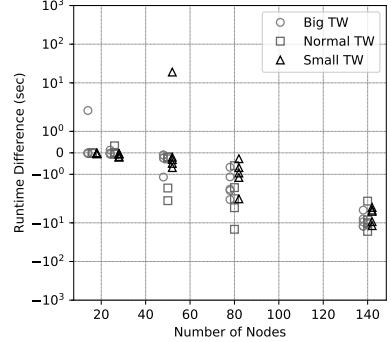


Fig. 10: Runtime difference of pure CP minus CP with MIP/CP LBBB starting solution.

We conducted the same experiments, inserting the MIP/CP LBBB first solution into the CP model as a warm start with the results shown in Figure 10. In most cases, with the given assignment of the warm start solution, the CP model performs slightly slower. Examination of the actual vehicle assignments of the first solutions showed that MIP/CP LBBB's assignment often clusters requests onto a few vehicles. When CP is warm-started with such solutions, it needs to backtrack and reassign many requests to different vehicles in order to prove optimality.

## 5.2 Search Space Reduction

The next set of experiments measures the impact of search space reduction of 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 (49).

$$\log(|P|) = \log(|D_{x_1}|) + \dots + \log(|D_{x_n}|) \quad (49)$$

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  $|D_{x_i}| = L_i - E_i - S_i + 1$ . We focus on the CP/CP LBBB model so as to not conflate the comparison with fundamentally different problem solving bases (e.g., back-propagation is less important for MIP solving).

Since the optimal solution for each dataset is already known, we compute 5 different lower bounds for each dataset that are 100%, 80%, 60%, 40%, and 20% of the 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 LBBB approach. Both approaches are then allowed to simply propagate and the size of the search space is calculated. Table 5 presents how many instances show search space reduction given the different lower bounds for both the CP and CP/CP LBBB.

Table 5: 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

		Lower Bound Percentage				
		TW Type				
		100%	80%	60%	40%	20%
CP	small	8	3	2	0	0
	normal	3	1	0	0	0
	big	0	0	0	0	0
CP/CP LBBB	small	1	1	0	0	0
	normal	0	0	0	0	0
	big	0	0	0	0	0

There are several instances that show 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 LBBB showing a poor propagation of the first solution quality to the entire search space.

## 6 Conclusion

Inspired by real-world problems, we define the Senior Transportation Problem, a problem encountered by organizations responsible for providing elder transportation. We show that it is a challenging combination of Pickup-and-Delivery with Time Windows, the Dial-a-Ride Problem, and the Team Orienteering Problem. In this paper, a formal problem definition for the STP was proposed, illustrating multiple constraints inspired from real life problems.

Five different approaches using mixed integer programming, constraint programming, logic-based Benders decomposition, and a construction heuristic are developed to solve the STP. Each method is tested on 75 instances from a generated dataset and 280 real-world instances from our industrial partner. Constraint programming proves to be the best performing approach on both problem sets in terms of the number of instances solved to proven optimality, faster runtime, and solution quality. An LBBB approach combining mixed integer programming and constraint programming achieves the second best performance, though substantially worse than the pure constraint programming model. Our subsequent analysis lends support to the hypotheses that the strong performance of the CP model stems from the ability to quickly find good solutions and then to use the bounds on those solutions to reduce the search space.

While our conclusion is that the current CP model is superior, we plan to try to improve the logic-based Benders models in order to further challenge the pure CP approach and, more importantly, develop a deeper understanding of the problem characteristics that favor constraint programming or decomposition approaches.

## References

1. Baklagis, D., Dikas, G., Minis, I.: 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)
2. Cordeau, J.F., Laporte, G.: The dial-a-ride problem: models and algorithms. *Annals of operations research* 153(1), 29 (2007)
3. Parragh, S.N., Doerner, K., Hartl, R.F.: A survey on pickup and delivery models: Part i: Transportation between customers and depot. *Journal für Betriebswirtschaft* 58(1), 21–51 (2008)
4. Parragh, S.N., Doerner, K.F., Hartl, R.F.: A survey on pickup and delivery problems: Part ii: Transportation between pickup and delivery locations. *Journal für Betriebswirtschaft* 58, 81–117 (2008)
5. Qiu, X., Feuerriegel, S., Neumann, D.: Making the most of fleets: A profit-maximizing multi-vehicle pickup and delivery selection problem. *European Journal of Operational Research* (2016), <http://www.sciencedirect.com/science/article/pii/S0377221716308244>
6. Refalo, P.: Impact-based search strategies for constraint programming. CP 3258, 557–571 (2004)
7. Ropke, S., Cordeau, J.F.: Branch and cut and price for the pickup and delivery problem with time windows. *Transportation Science* 43(3), 267–286 (2009), <http://dx.doi.org/10.1287/trsc.1090.0272>
8. Vansteenwegen, P., Souffriau, W., Van Oudheusden, D.: The orienteering problem: A survey. *European Journal of Operational Research* 209(1), 1–10 (2011)