Modelling and Solving the Senior Transportation Problem

Chang Liu(✉), Dionne M. Aleman, and J. Christopher Beck(✉)

Department of Mechanical and Industrial Engineering, University of Toronto, Toronto, ON M5S 3G8, Canada
{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. Building on the vehicle routing literature, we develop 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 s. 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 waiting lists. 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

© Springer International Publishing AG, part of Springer Nature 2018
W.-J. van Hoeve (Ed.): CPAIOR 2018, LNCS 10848, pp. 412–428, 2018.
https://doi.org/10.1007/978-3-319-93031-2_30

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 analyses 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 to optimality.

2 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 $\mathcal{K} = \{1, \ldots, |\mathcal{K}|\}$ 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 a depot 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 its starting depot during the first interval, perform all pickup and delivery requests assigned to it, and arrive at its ending depot during the second interval. Furthermore, vehicles differ in capacity, with each vehicle $k \in \mathcal{K}$ associated with a maximum capacity C_k.

Let $\mathcal{R} = \{1, \ldots, |\mathcal{R}|\}$ 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 client 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, $[i_{k^+}, \ldots, i_{k^-}]$ 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.

3 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) [10]. Alternatively, the routing and assignment of requests can be seen as a Pickup and Delivery Problem with Time Windows (PDPTW) [5,6] or a Dial-a-Ride Problem (DARP) [2]. In addition to minimizing total travel cost in the classical DARP, Cordeau and Laporte [2] noted that there can be other objectives, such as maximizing the number of fulfilled demands or overall quality of service, but did not provide any formulation or references. The PDPTW has been solved to optimality for loosely constrained instances of sizes up to 100 requests [9] 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. [7] 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 times for clients, and heterogeneous fleets.

4 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.

4.1 Mixed Integer Programming

In Fig. 1, we present a MIP formulation adapted from the PDPTW formulation of Ropke and Cordeau [9]. 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 $i \in \mathcal{V}$. They are non-negative and less than or equal to the vehicle capacity C_k.

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

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

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

same vehicle. In Constraint (7), the travel time and service time of visited locations are enforced. Constraints (8) and (9) make sure that each location that is visited must be visited within its time window. Constraint (10) imposes that pickup locations must precede delivery locations. 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 location.

4.2 Constraint Programming

The CP formulation (Fig. 2) employs optional interval variables [4] that are 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

$$\max \sum_{r \in \mathcal{R}} \left(W_r \times \texttt{PresenceOf}(x_{i_{r+}})\right) \tag{16}$$

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

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

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

$$\texttt{StartOf}(x_{i_{r-}}) - \texttt{EndOf}(x_{i_{r+}}) \leq F \quad \forall r \in \mathcal{R} \tag{20}$$

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

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

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

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

$$\texttt{NoOverlap}(u_k, T) \quad \forall k \in \mathcal{K} \tag{25}$$

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

regardless of 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 $\bar{X}_{k,i}$ are transpositions of each other (i.e., $X_{i,k} = \bar{X}_{k,i}$), and link the x_i variables to vehicles through the use of the `Alternative` constraint. 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. The `Alternative` constraint in Constraint (17) indicates that if a variable (x_i) is present, then exactly one variable in the set of variables X_i (a vector of variables $X_{i,k}$) can be present, ensuring that at most one vehicle can visit location i. In Constraint (18), the `Before` constraint ensures that each pickup location is visited before its corresponding delivery location. 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 difference between the end time of a delivery location variable and the start time of the respective pickup location variable must be less than the maximum ride time and is enforced through Constraint (20). In Constraint (21), the *cumul* function $v_{k,i}$ is defined such that for each vehicle k, the variable changes by the load size of location i, Q_i, at the start of the location variable of vehicle k ($\bar{X}_{k,i}$) where the size is positive for a pickup and negative for a delivery. Constraint (22) enforces that the sum of the load variables does not exceed the capacity of the vehicle. Constraints (23) and (24) indicate that each route must start at its associated start depot and end at its associated end depot. The CP model uses the `NoOverlap` global constraint (25) to prevent sub-tours on each route; it specifies that all present

$$
\begin{aligned}
\max \quad & \sum_{r \in \mathcal{R}} \sum_{k \in \mathcal{K}} (W_r \times \varphi_{k,r}) && && (26)\\
\text{s.t.} \quad & \sum_{k \in \mathcal{K}} y_{k,i} \leq 1 && \forall i \in \mathcal{N} && (27)\\
& \zeta_r = S_{i_{r+}} + T_{i_{r+}, j_{r-}} + S_{i_{r-}} && \forall r \in \mathcal{R} && (28)\\
& Q_r \times \varphi_{k,r} \leq P_k && \forall k \in \mathcal{K}, r \in \mathcal{R} && (29)\\
& (E_{i_{r+}} + \zeta_r) \times \varphi_{k,r} \leq L_{i_{k-}} && \forall k \in \mathcal{K}, r \in \mathcal{R} && (30)\\
& (E_{i_{k+}} + \zeta_r) \times \varphi_{k,r} \leq L_{i_{r-}} && \forall k \in \mathcal{K}, r \in \mathcal{R} && (31)\\
& \sum_{i \in \mathcal{N}} (y_{k,i} \times \mathcal{T}_i + S_i) + \mathcal{T}_{i_{k+}} + S_{i_{k+}} && && \\
& \qquad \leq L_{i_{k-}} - E_{i_{k+}} && \forall k \in \mathcal{K} && (32)\\
& y_{k,r} = y_{k,r+|\mathcal{R}|} = \varphi_{k,r} && \forall k \in \mathcal{K}, r \in \mathcal{R} && (33)\\
& y_{k,i}, \varphi_{k,r} \in \{0,1\} && \forall k \in \mathcal{K}, i \in \mathcal{N}, r \in \mathcal{R} && (34)\\
& \textit{Benders Cuts} && &&
\end{aligned}
$$

Fig. 3. A MIP model for the LBBD master problem of the STP.

interval variables on the sequence variable u_k must not overlap in operation times while considering the transition time between all locations defined through the transition distance matrix T.

4.3 Logic-Based Benders Decompositions

For the LBBD approaches [3], we decompose 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.

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

Fig. 4. A CP model for the LBBD master problem of the STP.

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, ζ_r is modelled in Constraint (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 Fig. 4 uses significantly fewer of variables than the full STP model in Fig. 2. 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 Sect. 4.2.

The objective and a number of constraints remain the same as in the full STP model (Fig. 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 allocates the requests, a sub-problem is created for each vehicle with at least two assigned requests.[1] 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 a proper subset of the assigned requests. The objective value of the sub-problem is then used in a Benders cut. With

[1] The master problem guarantees a solution for a vehicle with only one request.

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

s.t.

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

$$\texttt{StartOf}(x_{i^-}) - \texttt{EndOf}(x_{i^+}) \leq F \qquad \forall i \in \mathcal{R}^* \tag{41}$$

$$v_i = \texttt{StepAtStart}(v_i, Q_i) \qquad \forall i \in \mathcal{N}^* \tag{42}$$

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

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

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

$$\texttt{NoOverlap}(u) \tag{46}$$

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

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 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^* in iteration h, denoted by $\mathcal{J}_{h,k}$, the objective value 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).

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

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

4.4 A Construction Heuristic

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

Since the objective function is to maximize the weight of served requests, it is reasonable to first schedule the requests that have the highest ratio of weight to length (i.e., W_r/ζ_r with ζ_r as defined in Constraint (28)). Furthermore, vehicles are sorted in ascending order of the size of their interval of availability 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 $\frac{W_r}{\zeta_r}$;
Sort $\mathcal{K}$ based on ascending 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 2 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 sizes;
      break;
    end
  end
end
Regroup all pieces of the same vehicle to make scheduled routes;

5 Experimental Results

In this section, we discuss the datasets used in our experiments and present the performance of the five approaches proposed above, including using the construction 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 @ 3.30 GHz, 16 GB RAM using a single thread and a 600 s runtime limit. The CP Optimizer solver is set to use its default search.

5.1 Datasets

We generated 75 random datasets and extracted 280 problem instances from real world data provided by a partnering organization. In the generated problem 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 normal distributions. Table 1 outlines the lower and upper bounds of each characteristic.

Table 1. Bounds on problem characteristics for generated datasets.

	Characteristic	Lower bound	Upper bound
Vehicle	Number of vehicles	2	20
	Capacity	2	6
	Start and end depot service time	2	16
Request	Number of requests	6	50
	Size	1	3
	Weight	1	5
	Pickup and delivery location service time	2	16
	Travel time	1	60
Time windows	Small	80	180
	Normal	180	360
	Big	600	900

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.

5.2 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 s, MIP/CP LBBD solved 71 (95%) instances with an average runtime of 21.78 s, CP/CP

LBBD solved 49 (65%) instances with an average runtime of 110.14 s, and MIP solved 35 (48%) instances with an average of 90.00 s. 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 heuristic start only improves MIP/CP LBBD a little while it has very minimal effects on CP. CP/CP LBBD 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 subproblems are solved.

Table 2. Number of instances solved to optimality, average runtime, and average optimality gap for generated datasets. The '*' indicates the heuristic found but did not prove optimal solutions.

Approach	# instances solved to optimality	% solved to optimality	Average runtime	Average optimality gap
Heuristic	45*	60.00%*	**0.01**	4.13%
MIP	35	46.67%	90.00	30.68%
MIP_H	52	69.33%	22.36	1.80%
CP	**75**	100.00%	1.02	**0.00%**
CP_H	**75**	100.00%	2.38	**0.00%**
MIP/CP LBBD	71	94.67%	21.78	0.15%
MIP/CP LBBD_H	73	97.33%	2.54	0.09%
CP/CP LBBD	49	65.33%	110.14	18.95%
CP/CP LBBD_H	61	81.33%	42.58	10.85%

Given the good performance of CP and MIP/CP LBBD, 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 s using the pure CP model while the MIP/CP LBBD could only solve 47 instances in 331.31 s.

The evolution of runtime of the CP model as the problem sizes of the real instances increase is shown in Fig. 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-h time limit. An additional 21 instances were solved to optimality but nine instances are still open. Thus Table 4 reports the

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 LBBD	47	16.79%	331.31	18.38%

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%	**18.38%**

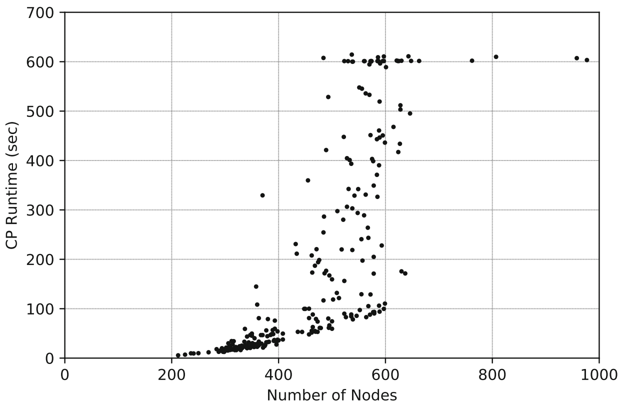

Fig. 6. CP runtime of real world instances over number of vertices.

solution quality relative to the best known solution for the real world datasets. The overall mean optimality gap for CP is 5.25% and 11.84% for MIP/CP LBBD.

6 Analysis

The strong results for CP compared to the LBBD approaches differ from much of the literature. Here, we explore three, non-mutually exclusive, hypotheses.

1. The default search of CP Optimizer is particularly suited to our problems.

2. The first feasible solutions found by the CP model are better and found more quickly than those found by the LBBD approaches.
3. Good solutions result in strong back-propagation from the lower-bound on the objective function, creating greater impact of search space reduction [8].

6.1 CP and Depth First Search

CP Optimizer's default search employs a combination of Large Neighbourhood Search (LNS) and Failure-directed Search (FDS) [11]. 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 s to 1.873 s, 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 s to 0.207 s.

The difference when 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.

6.2 First Solution Quality and Time

We recorded the time to find the first feasible solution and its quality for the CP model and both LBBD approaches on the generated dataset. The objective value, z', is compared to the known optimal solution, z^* via the optimality gap 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.

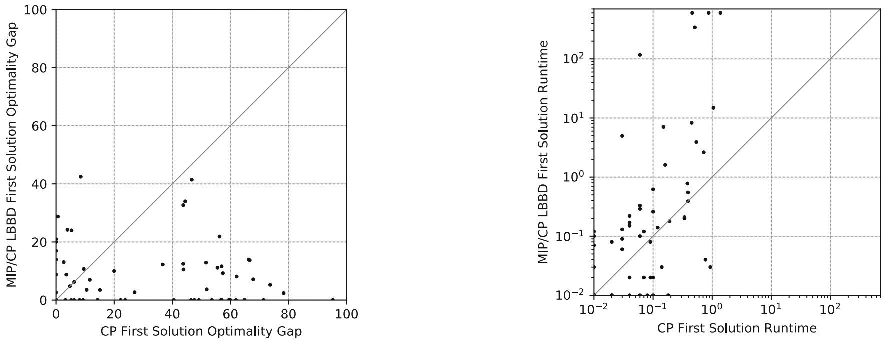

Fig. 7. First solution quality of MIP/CP LBBD compared to CP.

For the LBBD approaches, the first feasible solution is often the actual optimal solution and therefore is usually better than the CP model. However, the time to find these solutions for the LBBD approaches is much longer.

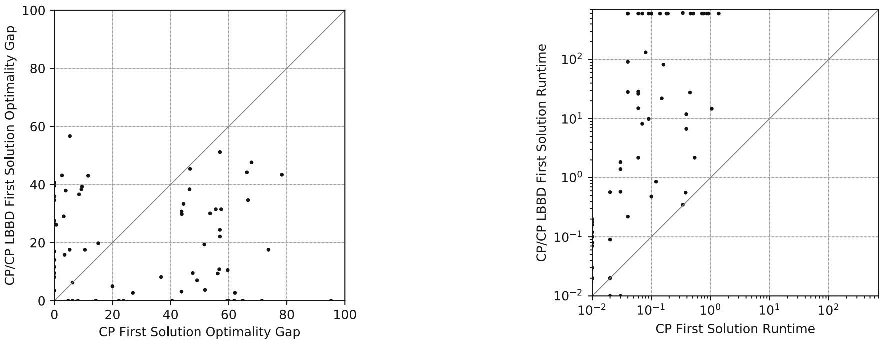

Fig. 8. First solution quality of MIP/CP LBBD compared to CP.

To further analyze the effect of the first solution, 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 run-time 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. For this experiment, the runtime does not include the time to compute the warm start solution. The results are shown in Fig. 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.

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

6.3 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 Eq. (49) [8].

$$\log(|P|) = \log(|D_{x_i}|) + \ldots + \log(|D_{x_n}|) \tag{49}$$

For interval variables, the domain size is simply the size of the interval minus the duration of the variable, or $|D_{x_i}| = L_i - E_i - S_i + 1$. For optional interval

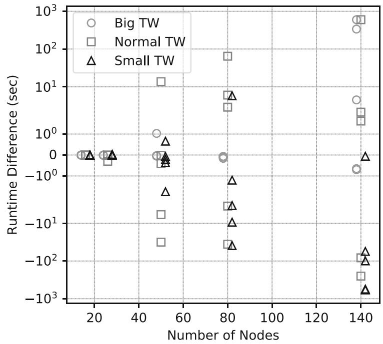

Fig. 9. Runtime difference of pure MIP/CP LBBD minus MIP/CP LBBD with CP starting solution.

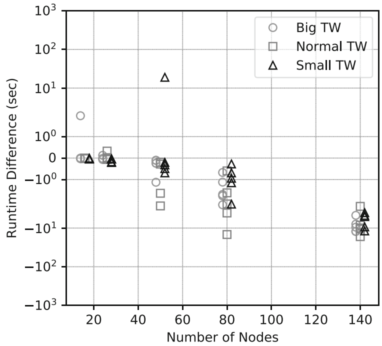

Fig. 10. Runtime difference of pure CP minus CP with MIP/CP LBBD starting solution.

variables, there is an additional boolean value to represent the presence of the variable, thus the domain size is multiplied by 2. We focus on the CP/CP LBBD model so as to not conflate the comparison with fundamentally different problem solving bases (e.g., back-propagation is less important for MIP solving).

From the known optimal solutions, we compute five 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 LBBD approach. The search space is calculated before and after propagation of the root node. Table 5 presents how many instances show search space reduction and the average percentage reduction *over those instance which showed non-zero reduction*, given the different lower bounds for both CP and CP/CP LBBD.

Table 5. Number of instances (out of 25 in each row) that show a reduction in search space and the average percentage reduction after applying the artificial lower bound. The average only includes instances with non-zero reduction.

	TW Type	Lower Bound Percentage				
		100%	80%	60%	40%	20%
CP	small	8 (9.53%)	3 (23.56%)	2 (34.55%)	0 (-)	0 (-)
	normal	3 (1.50%)	1 (0.30%)	0 (-)	0 (-)	0 (-)
	big	0 (-)	0 (-)	0 (-)	0 (-)	0 (-)
CP/CP LBBD	small	1 (5.61%)	1 (5.61%)	0 (-)	0 (-)	0 (-)
	normal	0 (-)	0 (-)	0 (-)	0 (-)	0 (-)
	big	0 (-)	0 (-)	0 (-)	0 (-)	0 (-)

There are several instances that show a search space reduction for CP, after applying a lower bound, indicating back-propagation. Only one instance demonstrated search space reduction for the CP/CP LBBD showing a poor propagation of the first solution quality to the entire search space.

The average percentage reduction should be interpreted carefully. Since we are taking the mean the over instances with non-zero reduction, it may increase even when the lower bound decreases due to fewer problems showing any reduction (i.e., a smaller denominator).

7 Conclusion

Inspired by a real-world problem, we define the Senior Transportation Problem (STP), 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 in 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 LBBD 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 feasible 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 at a deeper understanding of the problem characteristics that favor CP 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-Oper. Res. **50**(3), 503–517 (2016)
2. Cordeau, J.F., Laporte, G.: The dial-a-ride problem: models and algorithms. Ann. Oper. Res. **153**(1), 29 (2007)
3. Hooker, J.N., Ottosson, G.: Logic-based benders decomposition. Math. Program. **96**(1), 33–60 (2003)
4. Laborie, P., Rogerie, J.: Reasoning with conditional time-intervals. In: FLAIRS Conference, pp. 555–560 (2008)

5. Parragh, S.N., Doerner, K., Hartl, R.F.: A survey on pickup and delivery models: part I: transportation between customers and depot. J. für Betriebswirtschaft **58**(1), 21–51 (2008)
6. Parragh, S.N., Doerner, K.F., Hartl, R.F.: A survey on pickup and delivery problems: part II: transportation between pickup and delivery locations. J. für Betriebswirtschaft **58**, 81–117 (2008)
7. Qiu, X., Feuerriegel, S., Neumann, D.: Making the most of fleets: a profit-maximizing multi-vehicle pickup and delivery selection problem. Eur. J. Oper. Res. **259**(1), 155–168 (2017)
8. Refalo, P.: Impact-Based Search Strategies for Constraint Programming. In: Wallace, M. (ed.) CP 2004. LNCS, vol. 3258, pp. 557–571. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-30201-8_41
9. Ropke, S., Cordeau, J.F.: Branch and cut and price for the pickup and delivery problem with time windows. Transp. Sci. **43**(3), 267–286 (2009)
10. Vansteenwegen, P., Souffriau, W., Van Oudheusden, D.: The orienteering problem: a survey. Eur. J. Oper. Res. **209**(1), 1–10 (2011)
11. Vilím, P., Laborie, P., Shaw, P.: Failure-Directed Search for Constraint-Based Scheduling. In: Michel, L. (ed.) CPAIOR 2015. LNCS, vol. 9075, pp. 437–453. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-18008-3_30