



Optimization models for pickup-and-delivery problems with reconfigurable capacities

Arnoosh Golestanian ✉ 

Department of Mechanical and Industrial Engineering, University of Toronto, Toronto, Canada

Giovanni Lo Bianco ✉ 

Department of Mechanical and Industrial Engineering, University of Toronto, Toronto, Canada

Chengyu Tao ✉ 

Department of Mechanical and Industrial Engineering, University of Toronto, Toronto, Canada

J. Christopher Beck ✉ 

Department of Mechanical and Industrial Engineering, University of Toronto, Toronto, Canada

Abstract

When a transportation service accommodates both people and goods, operators sometimes opt for vehicles that can be dynamically reconfigured for different demands. Motivated by air service in remote communities in Canada's north, we define a pickup-and-delivery problem in which aircraft can add or remove seats during a multi-stop trip to accommodate varying demands. Given the demand for people and cargo as well as a seat inventory at each location, the problem consists in finding a tour that picks up and delivers all demand while potentially reconfiguring the vehicle capacity at each location by adding or removing seats. We develop a total of six models using three different approaches: constraint programming, mixed integer programming, and domain-independent dynamic programming. Our numerical experiments indicate that domain-independent dynamic programming is able to substantially outperform the other technologies on both solution quality and run-time on a set of randomly generated instances spanning the size of real problems in northern Canada.

2012 ACM Subject Classification Computing methodologies → Modeling methodologies

Keywords and phrases Pickup and delivery, Dial-a-ride problem, Optimization

Digital Object Identifier 10.4230/LIPIcs.CP.2023.34

Funding This research was supported by the Natural Sciences and Engineering Research Council of Canada.

1 Introduction

Pickup-and-delivery problems involve using vehicles to transport goods and/or passengers from a set of origins to a set of destinations on a given transportation network [1]. A typical pickup-and-delivery problem such as the Pickup and Delivery Traveling Salesperson Problem (PD-TSP) includes a one or more vehicles, requests with different pickup and delivery locations, and an objective to find a minimum-cost tour (or set of routes) that visit(s) each pickup location before its corresponding delivery location [4]. There has been substantial research literature on pickup and delivery problems over the past several years (e.g., [19, 21]) motivated, in part, by global efforts to reduce transportation-related carbon emissions [16]. Many variations of such problems have been proposed and studied in the operations research literature. For example, some problems include handling costs when an item is loaded or unloaded depending on the position of the item in the vehicle [24] and some include subsets of requests that cannot be in a vehicle at the same time [5].

In this paper, we propose and study a novel variation of PD-TSP: requests can include both goods (cargo) and passengers and the vehicle has a capacity that can be adjusted



© Arnoosh Golestanian, Giovanni Lo Bianco, Chengyu Tao, and J. Christopher Beck; licensed under Creative Commons License CC-BY 4.0

29th International Conference on Principles and Practice of Constraint Programming (CP 2023).

Editor: Roland H. C. Yap; Article No. 34; pp. 34:1–34:17

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

44 en-route depending on the request and equipment stored at locations in the network. The
 45 problem is motivated by a real transportation problem faced by air services in northern
 46 Canada. Since many communities in this region are reachable only by air during some parts
 47 of the year, their access to basic human needs such as fresh food and healthcare services is
 48 limited. The need for air transportation combined with the relatively small populations and
 49 lack of resources led northern air services to adopt the practice of transporting both cargo
 50 and passengers on the same flights. The vehicles are aircraft with removable seats, allowing
 51 staff to either remove passenger seats and store them at airports to transport more cargo or
 52 add additional seats, previously stored at airports, to carry more passengers. The problem,
 53 which we call the Pickup-and-Delivery with Seat Replacement Problem (PD-SRP), therefore
 54 requires finding the shortest tour delivering all goods and passengers from their origins to
 55 destinations without exceeding aircraft capacity but allowing seats to be removed from or
 56 added to aircraft at each location, subject to seat availability and total aircraft capacity.

57 To solve the PD-SRP, we developed three types of optimization models: one Constraint
 58 Programming (CP) model, three Mixed Integer Programming (MIP) models, and two Domain-
 59 Independent Dynamic Programming (DIDP) [14] models. We compare their performance
 60 on randomly generated instances based on the size of the problem in Canada's north,
 61 demonstrating that both of the DIDP models outperform the CP and MIP models in terms
 62 of the number of instances solved and proved optimal, solution quality, and solve time.

63 **2 Related Works**

64 Reconfigurable capacity is a general term in the transportation literature, typically indicating
 65 that vehicle capacity can be changed at some cost and/or limited by some constraints [22, 23].
 66 Other terms such as multi-compartment vehicle or multi-purpose vehicle are used to convey
 67 a similar meaning [20, 8]. We review the vehicle routing and dial-a-ride problems literature
 68 for studies that considered adjustable vehicles.

69 *Vehicle Routing Problems (VRP)*: The Vehicle Routing Problem and its many variations
 70 have been studied extensively over the past 50 years [18]. The idea of adjusting the vehicle
 71 to handle different types of demand has been studied in multi-compartment vehicle routing
 72 problems [20]. For example, Henke et al. [9] studied how to split the capacity of a truck
 73 into different compartments to maintain the separation of different colors of recycled glass.
 74 Similarly, for grocery distribution, different temperature-sensitive products can be transported
 75 on the same truck with multiple compartments [11]. In both of these problems, a vehicle's
 76 capacity configuration is fixed for its entire route and cannot be modified during the trip.

77 *Dial-a-Ride Problems (DARP)*: In the Dial-a-Ride Problem a transportation request
 78 takes the form of pickup and delivery location pair and the service provider utilizes its fleet
 79 of vehicles to fulfill the requests while minimizing a cost function that typically includes
 80 some travel distance component [10]. Some variants include a reconfigurable vehicle capacity
 81 to serve the needs of different users: those who use seats or those who use wheelchairs
 82 [23]. Some of the vehicle seats can be folded and stored inside the vehicle to make room
 83 for passengers in wheelchairs. Unlike this problem, the seats of the vehicle in the PD-SRP
 84 cannot be stored on the aircraft without occupying cargo capacity and are instead detached
 85 and stored at the airports.

86 Hatzenbühler et al. [8] studied a multi-purpose pickup and delivery problem that can
 87 deliver passengers or cargo by exchanging the module of the vehicle at a depot or special
 88 service site. Each vehicle includes a removable module and a fixed platform such that
 89 changing modules modifies the ability of the vehicle from only carrying cargo to only carrying

90 passengers and vice versa. Compared to problems with conventional solo-purpose vehicles,
 91 requests can be served with a fewer vehicles but at the expense of adding extra service sites
 92 and visits. We can view the core multi-purpose pickup and delivery problem as a special
 93 case of PD-SRP where the seat exchange decisions must be all-or-none: either all seats are
 94 removed to maximize cargo space or all seats are installed to maximize passenger capacity.

95 **3 Problem Definition**

96 In PD-SRP, we are given n requests, each potentially requiring the transportation of cargo
 97 and passenger demands. Let $V = P \cup D$ where $P = \{v_1, \dots, v_n\}$ is the set of pickup
 98 locations and $D = \{v_{n+1}, \dots, v_{2n}\}$ is the set of delivery locations. We assumed that cargo is
 99 shipped in unit-sized boxes, each having the same weight and volume. Although, in reality
 100 cargo is shipped in various shapes and weights, incorporating four-dimensional packing
 101 (i.e., combining volume and weight) would substantially complicate the problem. Therefore,
 102 similar to approximations done in practice by airlines (e.g., standard weight per passenger),
 103 we opted for this simplification.

104 Each request i includes picking up \hat{q}_i boxes of cargo and $\hat{\pi}_i$ passengers from location
 105 v_i and delivering them to location v_{n+i} . Thus, the demand of the corresponding delivery
 106 location has an equal magnitude negative value (i.e., $-\hat{q}_i = \hat{q}_{i+n}$, $-\hat{\pi}_i = \hat{\pi}_{i+n}$, $\forall i \in P$). Note
 107 that this representation can model more complex patterns (e.g., requests that share pickup
 108 or delivery locations but not both) by copying locations for each unique pickup-delivery pair.

109 When an aircraft is at its maximum seat capacity, it has \hat{S} seats and can carry \hat{C} boxes
 110 of cargo. By removing a seat, L boxes of cargo capacity are added to the aircraft. Therefore,
 111 the maximum cargo capacity when removing all the seats is $K = \hat{S}L + \hat{C}$. Each location i
 112 starts with S_i^0 stored seats and therefore the aircraft can add at most $\min(\hat{S}, S_i^0)$ seats or
 113 remove at most \hat{S} seats when visiting location i . There is no maximum number of seats that
 114 can be stored at a given location.

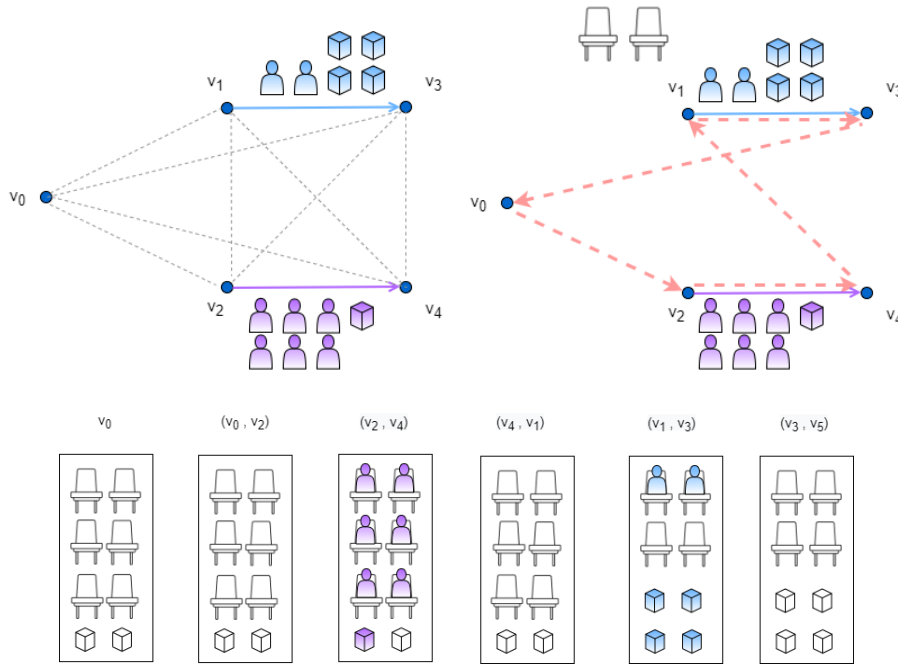
115 In order to represent the problem as a path, two nodes are assigned to the depot:
 116 v_0 is the start node and v_{2n+1} is the end node. For modeling purposes we define sets
 117 $V_{N+1} = V \cup \{v_{2n+1}\}$, $V_0 = V \cup \{v_0\}$ and $V_{0,N+1} = V \cup \{v_0, v_{2n+1}\}$. Therefore, the problem
 118 is defined on graph $G = (V_{0,N}, A)$ where $A = \{(i, j) | i, j \in V_{0,N+1}, i \neq j\}$ with each arc
 119 having an associated distance, d_{ij} . The vehicle is initially at the depot v_0 with a cargo and
 120 passenger capacity of \tilde{C}_0 and \tilde{S}_0 where $\tilde{C}_0 = K - L\tilde{S}_0$ and $\tilde{S}_0 \leq \hat{S}$, respectively, and must
 121 finish the trip at depot v_{2n+1} . We assume that the start and end nodes are not the pickup
 122 or delivery location of any requests. Again, this assumption is not limiting as such requests
 123 can be represented by adding extra nodes at the same location as the start and end nodes.

124 In PD-SRP we aim to minimize the travel distance while deciding how many seats to
 125 add or remove at each location to fulfill all the requests while respecting capacities. The
 126 PD-SRP is NP-hard because if we fix the seat decisions and set all the demands to zero, the
 127 problem can be reduced to TSP which is known to be NP-hard [13].

128 An instance of this problem can be seen in Figure 1. The optimal tour is shown in pink,
 129 and the seat icon near each vertex represents the number of seats stored at the corresponding
 130 base. The optimal tour for this instance is $(v_0, v_2, v_4, v_1, v_3, v_5)$ with two seats left at v_1 .

131 **4 Methods**

132 We develop six models for the PD-SRP using constraint programming (CP), mixed integer
 133 programming (MIP), and domain independent dynamic programming (DIDP). One of the



■ **Figure 1** Example of an PD-SRP instance with 2 requests. The optimal tour is shown by dotted pink edges. In the aircraft configuration, white seats and boxes show the current passenger and cargo capacity, respectively. The colored seats and boxes show the corresponding cargo and passenger requests that are picked up.

134 MIP models solves a restricted version of the PD-SRP and is used to warm-start the CP
 135 model and the two other MIP models. In this section, we describe each of the models in
 136 detail.

137 4.1 A Constraint Programming Model (CP)

138 Our CP model equates distance and time and, thus, uses a one-machine scheduling approach
 139 where jobs correspond to the visits and the setup times between two consecutive jobs
 140 correspond to the distance between two locations. The model uses $|V_{0,2n+1}|$ interval variables
 141 x_i that represent visits to each location, and a sequence variable, π , that constrains interval
 142 variables to form a sequence with an extra end node representing the return to the depot.
 143 The size of the interval variable is 0 because there is no service time associated with the
 144 visits. For every location $i \in \{0, \dots, 2n\}$, variable s_i is introduced to represent the number of
 145 seats that are added or removed. The formulation of the CP model is presented in Figure 2.
 146 Note that CP model is written in CP Optimizer language.

147 The objective function is the minimization of the total distance traveled by the aircraft.
 148 $\text{EndOf}(x_{2n+1})$ corresponds to the end-point of the last interval variable in the sequence
 149 variable π : the time (i.e., total distance travelled) when the aircraft returns to the depot.
 150 Constraint (1a) ensures that each pair of consecutive interval variables is scheduled with a
 151 transition time equal to at least the required travel distance between the two corresponding
 152 locations. Constraint (1b) enforces that the pickup location of each request is visited before
 153 the delivery location. Constraint (1c) specifies that the aircraft begins and ends at the start
 154 and end depot locations.

155 We used three cumulative functions to represent the following values that are potentially

$$\begin{aligned}
& \min \text{EndOf}(x_{2n+1}) && \text{(CP)} \\
& \text{s.t. NoOverlap}(\pi, \{d_{i,j} : (i,j) \in A\}) && \text{(1a)} \\
& \quad \text{EndBeforeStart}(x_i, x_{n+i}) && \forall i \in \{1, \dots, n\} \text{ (1b)} \\
& \quad \text{First}(\pi, x_0), \text{Last}(\pi, x_{2n+1}) && \text{(1c)} \\
& \quad C = \text{StepAt}(x_0, \tilde{C}_0) + \sum_{i=0}^{2n} \text{StepAtStart}(x_i, -\hat{q}_i - L \cdot s_i) && \text{(1d)} \\
& \quad C \geq 0 && \text{(1e)} \\
& \quad H = \text{StepAt}(x_0, \tilde{S}_0) + \sum_{i=0}^{2n} \text{StepAtStart}(x_i, -\hat{\pi}_i + s_i) && \text{(1f)} \\
& \quad H \geq 0 && \text{(1g)} \\
& \quad S = \text{StepAt}(x_0, \tilde{S}_0) + \sum_{i=0}^{2n} \text{StepAtStart}(x_i, s_i) && \text{(1h)} \\
& \quad S \leq \hat{S} && \text{(1i)} \\
& \quad x_i : \text{intervalVar}(0) && \forall i \in V_{0,2n+1} \text{ (1j)} \\
& \quad s_i : \text{integerVar}(-\hat{S}, \min(\hat{S}, S_i^0)) && \forall i \in V_0 \text{ (1k)} \\
& \quad \pi : \text{sequenceVar}(x_0, \dots, x_{2n+1}) && \text{(1l)}
\end{aligned}$$

■ **Figure 2** The CP Model for the PD-SRP.

156 changed by each interval variable (aircraft visits): available cargo space, number of empty
157 seats, and the total number of seats. In particular, cumulative functions (1d) and (1f) are
158 used to represent the available passenger and cargo space as the trip proceeds. H represents
159 the number of empty seats in the aircraft (i.e., the available passenger space) and C represents
160 the available cargo space. Before the trip starts, $K = H + C$ and, if there are \hat{S}_0 seats in
161 the aircraft at the start, $H = L \cdot \hat{S}_0$. The expression $\text{StepAtStart}(var, impact)$ specifies
162 the change (increment or decrement) to the cumulative function at the start of an interval
163 variable. The available cargo space C will decrease as cargo and seats are picked up, therefore
164 we use $\text{StepAtStart}(x_i, -\hat{q}_i - L \cdot s_i)$ to represent the changes to available cargo space at
165 each location $i \in \{1, \dots, 2n\}$. The available passenger space will decrease when cargo is
166 picked up, while increasing when adding seats as represented by $\text{StepAtStart}(x_i, -\hat{\pi}_i + s_i)$
167 at each location $i \in \{0, \dots, 2n\}$. The cumulative function S is introduced in constraint (1h)
168 to describe the change of the total number of seats in the aircraft. S will change with the
169 number of seats being added or removed as represented by s_i . Constraint (1i) restricts the
170 total number of seats by the maximum seat capacity \hat{S} . In constraint (1k), the domain of s_i
171 is $[-\hat{S}, S_i^0]$ reflecting the range of the number of seats that the aircraft can remove or add at
172 location i .

173 It should be noted that every interval variable contributes to the cumulative constraint,
174 which means that these bounds are maintained throughout the sequence. Therefore, we do
175 not need to have a separate cumulative function for every location of the tour.

176 4.2 Mixed Integer Programming Models

177 In this section, we describe three MIP models motivated by existing models for pickup and
 178 delivery problems. The first two models exactly represent the PD-SRP and therefore admit
 179 optimal solutions. The final model is a restriction of the PD-SRP problem that can be used
 180 to quickly find a feasible solution and, therefore, an upper bound for the PD-SRP. In our
 181 experiments, we investigate the use of this restricted model to warm-start the CP model and
 182 two other MIP models.

183 4.2.1 Two-indexed Location-Based MIP (MIIP_{loc})

184 We propose a two-indexed location-based MIP model for PD-SRP (MIIP_{loc}) based on a model
 185 for an existing pickup and delivery variant [7]. In MIIP_{loc} , x_{ij} is a binary variable that is 1 if
 186 arc $(i, j) \in A$ is traveled and is 0 otherwise. Non-negative continuous variables τ_i , u_i , and
 187 y_i represent the distance, available cargo space, and empty seats, respectively, on arrival at
 188 vertex $i \in V_{0,N+1}$. As above, let variable s_i be the number of seats that are added ($s_i > 0$) or
 189 removed ($s_i < 0$) at location i . Finally, let π_i and q_i be the number of passengers and boxes
 190 of cargo on the aircraft on arrival at vertex $i \in V_{0,N+1}$, respectively. The MIIP_{loc} model is
 191 shown in Figure 3.

192 The objective function minimizes the total distance traveled. Constraint (2a) ensures that
 193 each customer is visited exactly once while constraint (2b) forces an arrival and departure
 194 at each non-depot vertex. Constraints (2c) and (2d) prevent the formation of the subtours,
 195 using Miller-Tucker-Zemlin (MTZ) constraints [17]. Constraint (2e) forces the aircraft to
 196 visit the pickup location of each commodity before the delivery location. Constraints (2f) and
 197 (2g) respectively ensure that the total number of seats before and after adding or removing
 198 new seats does not exceed the passenger capacity. Similarly, constraint (2h) ensures that the
 199 total number of passengers on the aircraft after fulfilling the demand of vertex i does not
 200 exceed the passenger capacity. Constraint (2i) enforces the relationship between u_i and y_i .
 201 Note that the left hand side of the constraint restricts the picked-up cargo and passengers to
 202 not exceed the aircraft capacity. Constraints (2j) and (2k) define the upper bound and lower
 203 bound on the available cargo space, respectively. Similarly, constraints (2l) and (2m) set the
 204 upper and lower bounds on the number of empty seats. Constraint (2n) ensures that the
 205 passenger demand is met at each location, while constraint (2o) does the same thing for the
 206 cargo demand. Constraint (2p) restricts the number of the seats that can be added based on
 207 their availability. The lower bound on the number of removed seats, when $s_i < 0$, is always
 208 the number of seats on the aircraft at the arrival of location i . Lastly, constraints (2q) - (2s)
 209 specify binary and continuous variable domains.

210 4.2.2 Three-indexed Rank-Based MIP (MIIP_{rank})

211 The three-indexed ranked-based MIP model for PD-SRP (MIIP_{rank}) is adapted from a model
 212 for the multi-commodity pickup and delivery traveling salesperson problem [3]. In MIIP_{rank} ,
 213 $z_{i,j}^t$ is a binary variable indicating that aircraft goes directly from location i to location j
 214 and location i is at position t of the tour, for $i, j \in V_{0,N+1}$, $i \neq j$, $t \in \{0, \dots, 2n+1\}$. Binary
 215 variable $y_{i,t}$ is 1 if location i is visited at position t of the tour, $i \in V_{0,N+1}$, $t \in \{0, \dots, 2n+1\}$
 216 and 0 otherwise. Variable s_t is the number of seats added or removed at the t 'th position of
 217 the tour, for $t \in \{0, \dots, 2n+1\}$, with a negative value corresponding to the number of seats
 218 removed. Variables w_t and u_t represent the empty seats and available cargo space on arrival
 219 at t 'th position of the tour, for $t \in \{0, \dots, 2n+1\}$. Finally, let π_t and q_t be the number

$$\begin{aligned}
\min \quad & \sum_{i \in V_0} \sum_{j \in V_{N+1}, i \neq j} d_{ij} x_{ij} && (\text{MIP}_{loc}) \\
& \sum_{j \in V_{N+1}} x_{ij} = 1 && i \in V_0 \quad (2a) \\
& \sum_{j \in V_0, j \neq i} x_{ji} - \sum_{j \in V_{N+1}, i \neq j} x_{ij} = 0 && i \in V \quad (2b) \\
& \tau_i + x_{ij} - |V|(1 - x_{ij}) \leq \tau_j && i \in V_0, j \in V_{N+1}, i \neq j \quad (2c) \\
& 1 \leq \tau_i \leq |V| && i \in V \quad (2d) \\
& \tau_i + 1 \leq \tau_{n+i} && i \in P \quad (2e) \\
& y_i + \pi_i \leq \hat{S} && i \in V_0 \quad (2f) \\
& y_i + \pi_i + s_i \leq \hat{S} && i \in V_0 \quad (2g) \\
& \pi_i + \hat{\pi}_i \leq \hat{S} && i \in V_0 \quad (2h) \\
& u_i + q_i + Ly_i + L\pi_i \leq K && i \in V_0 \quad (2i) \\
& u_j \leq u_i - Ls_i - \hat{q}_i x_{ij} + (2K)(1 - x_{ij}) && i \in V, j \in V_{N+1}, i \neq j \quad (2j) \\
& u_j \geq u_i - Ls_i - \hat{q}_i x_{ij} - (2K)(1 - x_{ij}) && i \in V, j \in V_{N+1}, i \neq j \quad (2k) \\
& y_j \leq y_i + s_i - \hat{\pi}_i x_{ij} + 2\hat{S}(1 - x_{ij}) && i \in V, j \in V_{N+1}, i \neq j \quad (2l) \\
& y_j \geq y_i + s_i - \hat{\pi}_i x_{ij} - 2\hat{S}(1 - x_{ij}) && i \in V, j \in V_{N+1}, i \neq j \quad (2m) \\
& \pi_j \geq \pi_i + \hat{\pi}_i x_{ij} - \hat{S}(1 - x_{ij}) && i \in V, j \in V_{N+1}, i \neq j \quad (2n) \\
& q_j \geq q_i + \hat{q}_i x_{ij} - K(1 - x_{ij}) && i \in V, j \in V_{N+1}, i \neq j \quad (2o) \\
& -y_i \leq s_i \leq \min(\hat{S}, \mathcal{S}_i^0) && i \in V \quad (2p) \\
& \tau_0 = \pi_0 = q_0 = 0, u_0 = \tilde{C}_0, y_0 = \tilde{S}_0 && (2q) \\
& x_{ij} \in \{0, 1\} && i \in V_0, j \in V_{N+1}, i \neq j \quad (2r) \\
& u_i, y_i, \pi_i, q_i, \tau_i \in \mathbb{R}^{0+}, s_i \in \mathbb{R} && i \in V_0, N+1 \quad (2s)
\end{aligned}$$

■ **Figure 3** The MIP_{loc} Model for the PD-SRP.

220 of passengers and boxes of cargo on arrival at t 'th position of the tour. The MIP_{rank} is
 221 presented in Figure 4.

222 The objective function minimizes the total travel distance. Constraints (3a) and (3b)
 223 ensure that tour positions are assigned to exactly one location and that each location is
 224 visited exactly once, respectively. Constraint (3c) calculates the number of empty seats just
 225 before visit t , where $\sum_{i=1}^n y_{i,t-1} \hat{\pi}_i$ is the number of passengers picked up at position $t-1$
 226 of the tour. Similarly, constraint (3d) calculates the available cargo space just before visit
 227 t , where $\sum_{i=1}^n y_{i,t} \hat{q}_i$ is the amount of cargo picked up at position t of the tour. Constraint
 228 (3e) states that each commodity is picked up before it is delivered. Constraint (3f) enforces
 229 the relationship between w_t and u_t . The left hand side of the constraint enforces that the
 230 picked-up cargo and passengers do not exceed the available aircraft capacity. Constraint
 231 (3g) ensures that there is always \hat{C} space available for cargo on the aircraft. From (3f)
 232 and (3g) we can conclude that $\pi_t + w_t \leq \hat{S}$: the total number of seats does not exceed
 233 the passenger capacity. Constraint (3h) ensures the feasibility of the number of seats to
 234 be added or removed. Constraints (3i) and (3j) calculate the number of passengers and
 235 boxes of cargo at each position of the tour, respectively. Constraints (3k) and (3l) enforce
 236 the relationship between y and z variables and, together with (3e) and (3m), prevent the
 237 formation of subtours in a MTZ fashion. Lastly, constraints (3n) - (3q) specify the domains
 238 of the variables.

239 4.2.3 Upper bound MIP Model (MIP_{UB})

240 Our preliminary experiments suggested that the CP and MIP models presented above
 241 struggled to find good feasible solutions. We, therefore, investigate the use of a third MIP
 242 model, designed to quickly find an upper bound on the PD-SRP by solving a restriction of
 243 the full problem. Such a model provides a heuristic solution as well as a potential warm-start
 244 solution for the complete models.

245 The upper bound model is obtained by over-constraining the original problem to require
 246 that a request must be delivered immediately after being picked up. The nodes in this
 247 problem include the start depot v_0 , the end depot v_{N+1} , and all the pickup nodes $P =$
 248 $\{v_1, \dots, v_n\}$. For modeling purposes we define sets $P_{N+1} = P \cup \{v_{N+1}\}$, $P_0 = P \cup \{v_0\}$
 249 and $P_{0,N+1} = P \cup \{v_0, v_{N+1}\}$. The delivery nodes are not explicitly included because each
 250 origin-to-destination trip takes place immediately after the visit to the pickup node with the
 251 total distance increased by both the travel to the pickup node and the travel between the
 252 pickup node and the delivery node.

253 The MIP_{UB} model is presented in Figure 5. Let $x_{i,j}$ be a binary variable indicating that
 254 the aircraft goes from the delivery location of the request i to the pickup location of request
 255 j . Let s_i be the number of seats in the aircraft right after visiting location i . Finally, let t_i
 256 be the position of location i on the tour. The solution returned by this model is likely to be
 257 sub-optimal for the PD-SRP.

258 The objective function minimizes the total distance traveled. The coefficient c_{ij} represents
 259 the total distance starting from the delivery location of request i , visiting the pickup location
 260 of request j , and then travelling to the delivery locations of request j . Request 0 is to travel
 261 from the depot to the pickup location of the first request. The delivery and pickup locations
 262 of request 0 are nodes v_0 and v_{2n+1} , respectively.

263 Constraints (4a) and (4b) ensure that each node is visited exactly once. Constraints
 264 (4c), (4f), and (4i) prevent the formation of subtours. Constraint (4d) describes seat changes
 265 when the aircraft visits a node and constraint (4e) requires that the space taken up by the
 266 seats in the aircraft must be less than or equal to the remaining space after picking up the

$$\begin{aligned}
\min \quad & \sum_{t=0}^{2n} \sum_{i \in V_{0,N+1}} \sum_{j \in V_{0,N+1}, j \neq i} d_{i,j} z_{i,j}^t && (\text{MIP}_{rank}) \\
& \sum_{i \in V_0} y_{i,t} = 1 && t \in \{0, \dots, 2n\} \quad (3a) \\
& \sum_{t=1}^{2n} y_{i,t} = 1 && i \in V \quad (3b) \\
& w_t = w_{t-1} + s_{t-1} - \sum_{i=1}^n y_{i,t-1} \hat{\pi}_i && t \in \{1, \dots, 2n+1\} \quad (3c) \\
& u_t = u_{t-1} - L s_{t-1} - \sum_{i=1}^n y_{i,t-1} \hat{q}_i && t \in \{1, \dots, 2n+1\} \quad (3d) \\
& \sum_{t=1}^n t y_{i,t} - \sum_{t=1}^n t y_{n+i,t} \leq -1 && i \in P \quad (3e) \\
& q_t + u_t + L w_t + L \pi_t \leq K && t \in \{0, \dots, 2n+1\} \quad (3f) \\
& q_t + u_t \geq \hat{C} && t \in \{0, \dots, 2n+1\} \quad (3g) \\
& -w_t \leq s_t \leq \min(\hat{S}, \sum_{i=1}^n \mathcal{S}_i^0 y_{i,t}) && t \in \{0, \dots, 2n+1\} \quad (3h) \\
& \pi_t = \pi_{t-1} + \sum_{i=1}^n y_{i,t-1} \hat{\pi}_i && t \in \{1, \dots, 2n+1\} \quad (3i) \\
& q_t = q_{t-1} + \sum_{i=1}^n y_{i,t-1} \hat{q}_i && t \in \{1, \dots, 2n+1\} \quad (3j) \\
& y_{i,t} - \sum_{j=0}^n z_{i,j}^t = 0 && i \in V_0, t \in \{0, \dots, 2n\} \quad (3k) \\
& y_{j,t} - \sum_{i=0}^n z_{i,j}^{t-1} = 0 && j \in V_{0,N+1}, t \in \{1, \dots, 2n+1\} \quad (3l) \\
& y_{0,0} = y_{2n+1,2n+1} = 1, y_{0,t} = 0 && t \in \{1, \dots, 2n\} \quad (3m) \\
& s_t \leq \hat{S}, w_t \leq \hat{S}, u_t \leq K && t \in \{0, \dots, 2n+1\} \quad (3n) \\
& u_0 = \tilde{C}_0, w_0 = \tilde{S}_0, \pi_0 = q_0 = 0 && (3o) \\
& y_{i,t} \in \{0, 1\}, z_{i,j}^t \in \{0, 1\} && i, j \in V_{0,N+1}, t \in \{0, \dots, 2n+1\} \quad (3p) \\
& u_t, w_t, \pi_t, q_t \in \mathbb{R}^{0+}, s_t \in \mathbb{R} && t \in \{0, \dots, 2n+1\} \quad (3q)
\end{aligned}$$

■ **Figure 4** A Three-Indexed MIP Model for the PD-SRP.

$$\begin{aligned}
 \min \quad & \sum_{i \in P_0} \sum_{j \in P_0, i \neq j} c_{i,j} x_{i,j} && (\text{MIP}_{UB}) \\
 \text{s.t.} \quad & \sum_{j \in P_0, i \neq j} x_{i,j} = 1 && \forall i \in P \quad (4a) \\
 & \sum_{i \in P_{N+1}, i \neq j} x_{j,i} - \sum_{i \in P_0, i \neq j} x_{i,j} = 0 && \forall j \in P \quad (4b) \\
 & t_i + x_{i,j} - |P|(1 - x_{i,j}) \leq t_j && \forall i \in P_{N+1}, j \in P_{N+1}, i \neq j \quad (4c) \\
 & s_j \leq s_i + (S_{i+n}^0 + S_j^0)x_{i,j} + |\hat{S}|(1 - x_{i,j}) && \forall i \in P_0, j \in P_{N+1}, i \neq j \quad (4d) \\
 & Ls_i + \hat{q}_i \leq K && \forall i \in P_{N+1} \quad (4e) \\
 & 1 \leq t_i \leq |P| && \forall i \in P \quad (4f) \\
 & s_i \leq \hat{S} && \forall i \in P \quad (4g) \\
 & s_i \geq \hat{\pi}_i && \forall i \in P \quad (4h) \\
 & t_0 = 0 && (4i) \\
 & \tilde{S}_0 \leq s_0 \leq \tilde{S}_0 + S_0^0 && (4j) \\
 & x_{i,j} \in \{0, 1\} && \forall i \in P \quad (4k) \\
 & t_i \in \mathbb{N}, s_i \in \mathbb{N} && \forall i \in P_{0,N+1} \quad (4l)
 \end{aligned}$$

■ **Figure 5** The Upper Bound MIP model for a restriction of PD-SRP.

267 cargo of the current request. Constraint (4g) restricts the number of seats to never surpasses
 268 the maximum number of seats allowed in the aircraft and constraint (4h) ensures that the
 269 number of seats in the aircraft never drops below the number of passengers to be picked up.
 270 Constraints (4j) - (4l) specify the domains of the variables.

271 4.3 Domain-Independent Dynamic Programming Models

272 Domain-Independent Dynamic Programming (DIDP) is a recently proposed methodology
 273 for solving combinatorial optimization problems by formulating the problem as state-based
 274 dynamic program (DP) and using a generic solver to solve it [14]. DP models are declaratively
 275 formulated in Dynamic Programming Description Language (DyPDL), a solver-independent
 276 modeling formalism for DP that is inspired by AI planning. In DyPDL, a model consists of
 277 the following:

- 278 ■ *state variables*: variables that take on numeric, set, or set-element values that define the
 279 states in the search space of the problem
- 280 ■ *target state*: the problem state for which the optimal value is to be computed by the
 281 recursive formulation
- 282 ■ *constants*: state-independent values
- 283 ■ *transitions*: decisions in the DP that move between states
- 284 ■ *base cases*: a set of conditions that define states that terminate the recursion
- 285 ■ *state constraints*: conditions that must be satisfied by all states
- 286 ■ *dual bound*: an optional lower (upper) bound on the objective function for minimization
 287 (maximization) problems.

288 We developed two DIDP models for the PD-SRP.

$$\begin{aligned}
& \text{compute } Z(V, 0, 0, 0, \tilde{S}_0, 0) && (\text{DIDP}_{2T}) \\
Z(U, i, q, \pi, s, \alpha) &= \begin{cases} d_{i, 2n+1} & \text{if } U = \emptyset, \alpha = 1 \\ \min_{\delta \in T(q, \pi, s, i)} Z(U, i, q + w_i, \pi + u_i, s + \delta, 1) & \text{if } U \neq \emptyset, \alpha = 0 \\ \min_{j \in R(U, i)} d_{i, j} Z(U \setminus \{j\}, j, q, \pi, s, 0) & \text{if } U \neq \emptyset, \alpha = 1 \end{cases} && (5a) \\
Z(U, i, q, \pi, s, \alpha) &\geq 0 && (5b) \\
T(i, q, \pi, s) &= \left\{ \delta \in [-s, \hat{S}_i] \mid q + w_i \leq K - (s + \delta)L \wedge \pi + u_i \leq s + \delta, \delta \in \mathbb{Z} \right\} && (5c) \\
R(U, i) &= \{j \in U \mid (i, j) \in A \wedge (j \notin D \vee p_j \notin U)\}. && (5d)
\end{aligned}$$

■ **Figure 6** The Two-transition DIDP Model (DIDP_{2T}) for PD-SRP.

289 4.3.1 A Two-transition DIDP Model (DIDP_{2T})

290 Our first DIDP model has two types of transition: one to represent adding or removing seats
291 and picking up or delivering cargo and passengers and a second to model moving the aircraft
292 to a different location. In the model, a state is a tuple $\langle U, i, q, \pi, s, \alpha \rangle$, which represents
293 the set of unvisited vertices, U , the current location, i , the cargo load, q , the number of
294 passengers, π , the number of seats, s , and a flag representing which type of transition to
295 apply, α . We set $\alpha = 1$ if we have finished pickup/delivery at a location to indicate that the
296 next transition should be to move the aircraft. Otherwise, $\alpha = 0$.

297 The DIDP_{2T} model is defined in Figure 6. We focus first on Eqs. (5c) and (5d), which
298 respectively define the possible seat changes and possible next locations at a location i .

299 Suppose that the number of seats at the current location i is increased by δ . Since there
300 are S_i^0 seats stored at each location initially, when the aircraft has s seats, at i we can add
301 at most $\min\{S_i^0, \hat{S} - s\}$ seats and remove at most s seats. For simplicity we will denote
302 $\hat{S}_i = \min\{S_i^0, \hat{S} - s\}$. Therefore, $\delta \in [-s, \hat{S}_i]$. Let numeric constants w_i and u_i be the net
303 change of cargo and passengers at location i , respectively. The cargo will be increased by
304 w_i , so the current cargo will become $q + w_i \leq K - (s + \delta)L$, the current space for cargo.
305 Similarly, the number of passengers will be $\pi + u_i \leq s + \delta$. Lastly, δ must only take integer
306 values. With these conditions, Eq. (5c) specifies the values of δ .

307 Consider visiting the next location, j , from current location i . To be a valid location to
308 visit next, j must be unvisited ($j \in U$), it must be connected by an edge in the graph to
309 i ($(i, j) \in A$), and it must be either a pickup location ($j \notin D$) or its corresponding pickup
310 location must have already been visited. If we let p_j be the pickup location for the request
311 whose delivery location is j , then this final condition is: $p_j \notin U$. Eq. (5d) represents the
312 candidate locations to visit next after current location i .

313 The objective function specifies the state for which the optimal cost needs to be computed:
314 the state where all pickup and delivery nodes are unvisited, the current location is the start
315 depot (v_0), the cargo and passenger loads are 0, the aircraft has \tilde{S}_0 seats, and the next
316 transition should be to move the aircraft ($\alpha = 0$). In Eq. (5a), the first line computes the
317 cost to return to the depot from node i , the second line describes the cost of adding or
318 removing δ seats at node i , and the third line describes the cost of visiting node j from i .
319 Note that when the aircraft is moved, the state variable α is set to 0 and if the decision
320 regarding seats is made in this transition, α is set to 1. Constraint (5b) is a dual bound for
321 the DIDP model which is optional but may be exploited by the solver.

$$\begin{aligned}
& \text{compute } Z(V, 0, 0, 0, \tilde{S}_0) && (\mathbb{DIDP}_{1T}) \\
Z(U, i, q, \pi, s) = & \begin{cases} d_{i, 2n+1} & \text{if } U = \emptyset \wedge \exists \delta \in T(i, q, \pi, s) \\ \min_{(\delta, j) \in T(i, q, \pi, s) \times R(U, i)} d_{i, j} + Z(U \setminus \{j\}, j, q + w_i, \pi + u_i, s + \delta) & \text{if } U \neq \emptyset \end{cases} && (6a) \\
Z(U, i, q, \pi, s) \geq 0 &&& (6b) \\
& \text{Eq. (5c), Eq. (5d).}
\end{aligned}$$

■ **Figure 7** The One-transition DIDP Model (\mathbb{DIDP}_{1T}) for PD-SRP.

322 4.3.2 A One-transition DIDP Model (\mathbb{DIDP}_{1T})

323 We present the \mathbb{DIDP}_{1T} model in Figure 7. In this model, instead of two types of transitions,
324 we define one type that performs the pickup/delivery and seat exchange at a location and
325 then moves the aircraft to a new location. A state is the same as in \mathbb{DIDP}_{1T} with the
326 exception of the α flag which is no longer necessary: $\langle U, i, q, \pi, s \rangle$. As a transition first picks
327 up or delivers cargo, passengers, and seats at the current location and then moves the aircraft
328 to the next location, each transition corresponds to selecting (δ, j) : δ is the number of picked
329 up seats and j is the next location to visit. The set of possible decisions at each state is
330 therefore $T(i, q, \pi, s) \times R(U, i)$ as defined in the second line of Eq. (6a).

331 The objective function of \mathbb{DIDP}_{1T} defines the state for which the optimal cost is to be
332 calculated. It is identical to the target state in \mathbb{DIDP}_{2T} with the removal of α . In Eq. (6a),
333 the first line describes the cost of returning to the depot from node i , and the second line
334 describes the cost of visiting node j from i . Note that the first line checks if there exists
335 some δ such that the capacity constraints on the cargo and the passengers are satisfied. If
336 there is no such δ , we assume $Z(\emptyset, i, q, \pi, s) = \infty$.

337 4.3.3 Model Sizes and Solver

338 In a DIDP model, we need to define all transitions that are applicable in a state. In \mathbb{DIDP}_{2T} ,
339 δ can take an integer in $[-\hat{S}, \hat{S}]$ depending on a state, so there are $2\hat{S} + 1$ candidates. We
340 have $|V_{N+1}|$ locations to visit. Thus, \mathbb{DIDP}_{2T} requires $2\hat{S} + 1 + |V_{N+1}|$ transitions to be
341 defined in total. In contrast, \mathbb{DIDP}_{1T} needs to define $(2\hat{S} + 1)|V_{N+1}|$ transitions but does
342 not have state variable α . An alternative perspective is that the two DIDP models make
343 different trade-offs between the maximum branching factor and solution length. \mathbb{DIDP}_{1T} has
344 a branching factor of at most $(2\hat{S} + 1)|V_{N+1}|$ at each state and a solution path length of
345 $|V_{N+1}|$. \mathbb{DIDP}_{2T} has a maximum branching factor that alternates between $2\hat{S} + 1$ and $|V_{N+1}|$
346 and a solution length of $2|V_{N+1}|$. The performance of a solver is affected by the number of
347 state variables, the branching factor, and the solution length.

348 We solve the DIDP models with a complete anytime beam search (CABS) solver [25, 15].
349 CABS is an anytime algorithm meaning that seeks to quickly find a feasible solution and
350 then to improve it in the remaining run-time. CABS employs beam search: a heuristic search
351 algorithm that maintains a fixed number, b (beam width), of best states when exploring the
352 search space. In CABS, beam search is performed iteratively with increasing the beam width
353 until a stopping condition is met. Due to the iteratively increasing beam width, CABS is a
354 complete algorithm [25].

5 Numerical Evaluation

5.1 Experimental Setup

We have developed six different models, i.e., CP, MIP_{loc} , MIP_{rank} , MIP_{UB} , $DIDP_{1T}$, $DIDP_{2T}$. For the experiment, we use MIP_{UB} to warm-start the MIP and CP models, producing three additional approaches: MIP_{loc_W} , MIP_{rank_W} and CP_W .

To implement and solve the models we used Python v3.8.0 and the corresponding Python interfaces to the solvers: Gurobi Optimizer 10.0.1 and gurobipy for MIP, CP Optimizer 22.1.0.0 and DOCplex for CP, and didppy 0.3.3 for DIDP.¹ Each run has a time limit of 600s. The machine used to run the experiment has Intel(R) Core(TM) i7-9700 8 core CPU @ 3.00GHz, 12MB cache, and memory of 31G.

The models are tested on randomly generated instances with sizes 4, 6, 8, 10, 12, 15, and 20 with 10 instances per size. The size of each instance is the number of requests, which is half of the number of locations. We generate problem instances randomly, approximately reflecting real-world problem size, aircraft capacity and configurations, and stored seats at each location. We fix the maximum number of seats in the aircraft $\hat{S} = 6$, the cargo-to-seat ratio $L = 100$, and the cargo capacity on a full-seat aircraft $\hat{C} = 200$. The number of seats in the aircraft start configuration, \tilde{S}_0 , is selected uniformly from $\{0, \dots, 6\}$ and the cargo capacity in the start configuration is $\tilde{C}_0 = 800 - 100\tilde{S}_0$. Similarly, the number of seats available at location i , S_i^0 , is set uniformly from $\{0, \dots, 6\}$, independently for each location. The (x, y) coordinates of every location are uniformly generated from $\{0, \dots, 100\}^2$.

We generate the passenger and cargo demand to ensure the existence of capacity-feasible solutions. For each request $i \in \{1, \dots, n\}$, there is a demand of $\hat{\pi}_i$ passengers and demand of \hat{q}_i kg cargo (i.e., \hat{q}_i/L units of cargo). We first define the total number of passengers and units of cargo as $\hat{K} = \hat{q}_i/L + \hat{\pi}_i$, and \hat{K} is uniformly generated from $\{1, \dots, \hat{S} + \hat{C}/L = 8\}$. The passenger request $\hat{\pi}_i$ is then selected uniformly from $\{0, \dots, \min(\hat{S}, \hat{K})\}$. Consequently, the cargo request is $\hat{q}_i = L(\hat{K} - \hat{\pi}_i)$.

To compare the models, we used the number of instances solved and proved optimal, the PAR10 score time [12] (i.e., mean run-time with 10 times the time limit used if no optimal solution was proved), and mean relative error (MRE).

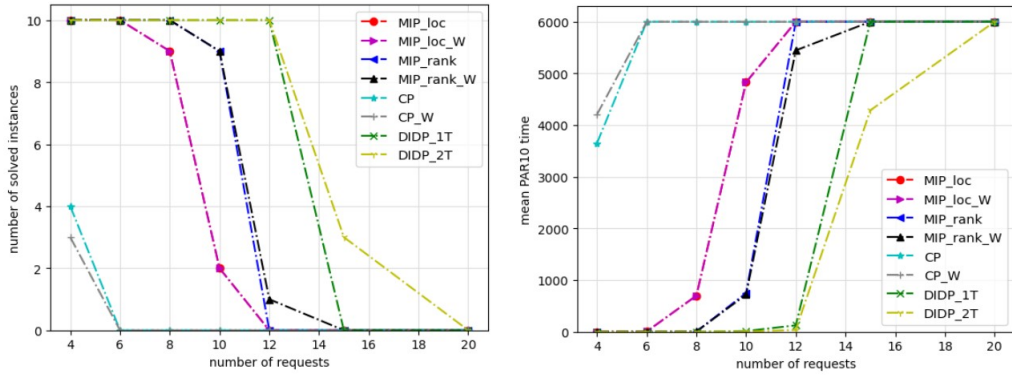
MRE compares the solution quality returned by each model. For an optimization problem let $obj_{t,m,i}$ be the objective value of the best solution achieved by time t of model m for instance i and let obj_i^* be the best-known objective value for that instance considering all the models. For the set of instances, \mathcal{I} , the relative error and mean relative error are computed in Eqs. (7) and (8). If a model did not find a feasible solution by a given time, the MIP_{UB} value is used to calculate a non-infinite measure.

$$RE(t, m, i) = \frac{obj_{t,m,i} - obj_i^*}{obj_i^*} \quad (7)$$

$$MRE(t, m) = \frac{1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} RE(t, m, i) \quad (8)$$

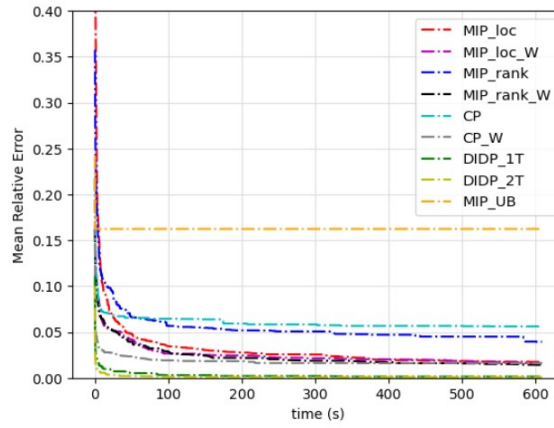
¹ <https://didp.ai>

34:14 Optimization models for PDPs with reconfigurable capacities



(a) Number of instances solved to optimality.

(b) Mean PAR10 time to solve instances.



(c) Comparison of MRE of all the models.

■ **Figure 8** Performance of MIP, CP and DIDP models.

392 5.2 Results

393 Figures 8a and 8b show the number of solved instances (i.e., proved infeasible or optimal) and
 394 mean PAR10 times for all the models. We do not include MIP_{UB} as it is incomplete, however
 395 for each model, its run-time is less than 0.02s. The run times for MIP_{loc_W} , MIP_{rank_W} ,
 396 and CP_W models, include the warm-start time.

397 The DIDP models solved all of the instances with 12 or fewer requests, with $DIDP_{2T}$
 398 performing slightly better than $DIDP_{1T}$ for instances of size 15 as it could solve three instances
 399 compared to none for $DIDP_{1T}$. Neither CP nor CP_W were able to solve any instances of size
 400 larger than 6 while the MIP models scaled up to size 10 or 12. There was one instance of
 401 size 4 that CP_W could not prove optimality, but CP could.

402 In terms of solution time, the DIDP models were the fastest and CP models were the
 403 slowest. For the MIP models, MIP_{rank_W} performed slightly better than MIP_{rank} in terms
 404 of both the number of solved instances and mean solution time. For one instance of size 12,
 405 MIP_{rank_W} proved optimality where MIP_{rank} could not.

406 The MRE graph is shown in Figure 8c. $DIDP_{2T}$ returns the best solutions and finds those
 407 best solutions within a few 10s of seconds. Up to 300s, CP_W outperformed MIP_{rank_W} ,
 408 MIP_{loc_W} , MIP_{loc} but after that point, their solution qualities are very similar. The solution
 409 qualities returned by CP are the worst after 100 seconds. However, the use of MIP_{UB} as a

410 warm start substantially improves CP quality especially for short run times. The performance
411 of the MIP_{loc} and MIP_{loc_W} models was very similar, however, the MIP_{loc_W} model returned
412 slightly better solution qualities than MIP_{loc} , especially before 200s. As we expected, the
413 solutions found by the incomplete MIP_{UB} are substantially worse than other models.

414 Overall, DIDP models performed better than MIP and CP, and in particular $DIDP_{2T}$
415 performed best in terms of the number of solved instances and average time to solve the
416 instances. We hypothesize that DIDP outperforms other models due to the combination
417 of tight capacity constraints and the precedence constraints induced by the pickup-and-
418 delivery structure. DIDP uses these constraints to prune many transitions and, thus, reduce
419 the search space. This result is consistent with previously observed behavior of DIDP on
420 constrained routing problems [15] and suggests an opportunity for research to understand
421 model characteristics that correlate with strong DIDP performance compared to other
422 optimization approaches.

423 6 Discussion

424 The contributions of this work are the introduction of a novel pickup-and-delivery problem
425 inspired by air services in northern Canada, the creation and evaluation of six optimization
426 models in three different frameworks, and the further demonstration that the recently
427 proposed domain-independent dynamic programming approach can out-perform incumbent
428 techniques in a model-and-solve paradigm.

429 While DP models are inherently state-based, the DIDP formalism provides a novel avenue
430 for constraint-based problem solving with connections to early ideas in CP (e.g., [6]). The
431 DIDP models for PD-SRP are unusual as DP models due to the extensive, constraint-based,
432 limitations on transitions (i.e., Eqs. (5c) and (5d)). While such limitations are key to strong
433 DP performance, they are typically procedurally implemented in a problem-specific DP
434 search algorithm. In DIDP, in contrast, constraint reasoning is used to prune transitions
435 based on the values of state variables rather than pruning variable domains based on partial
436 assignments. We believe that understanding this difference and developing constraint-based
437 reasoning for this context is a fruitful research direction for CP.

438 Our study has a number of limitations and opportunities for further research:

- 439 ■ In the definition of PD-SRP, we discretized cargo into identical boxes with one size
440 dimension (i.e., weight). In reality, cargo can take many forms from boxes of different
441 sizes and weights to baggage in various forms. Minimally, the volume of cargo needs to
442 be represented. More generally, the problem should address the four-dimensional (i.e.,
443 volume plus weight) packing of heterogeneous cargo.
- 444 ■ We made the assumption that passengers do not have travel time restrictions. However,
445 as a potential avenue for future research it would be interesting to incorporate additional
446 constraints regarding how long a single passenger can be stowed in the aircraft or how
447 long they can wait to be picked up.
- 448 ■ As is common in OR literature on transportation problems, our objective function is the
449 minimization of the travel distance. A more realistic objective would represent aspects
450 such as time and fuel consumption as well as handling and storage costs for seats.
- 451 ■ Most airlines run regular services with defined timetables and routings. Preliminary work
452 indicates that determining seat exchanges is an easy problem when routes are decided.
453 If this result bears out, there are two implications. First, we may have tools to deal
454 with harder aspects of the real world problem including multiple aircraft, uncertain and
455 dynamically changing demand (e.g., due to extreme weather in Canada's north), and

456 strategic decisions about timetable creation, seat inventory, and aircraft capacities. Second,
 457 even with the version of PD-SRP presented here, we may be able to scale by exploiting
 458 the “easy” seat exchange part of the problem through Benders decomposition [2].

459 ■ Although, in this study, our focus was to design simple models that can be used “off the
 460 shelf”, it is interesting to investigate sophisticated custom-constraint CP models in the
 461 future development of this work to see if they outperform the currently developed MIP
 462 and CP models.

463 **7 Conclusion**

464 This paper studied a novel pickup and delivery transportation problem with reconfigurable
 465 capacities, a problem inspired by air service in northern Canada. We defined the problem
 466 formally and developed six models in three different modeling formalisms: constraint pro-
 467 gramming, mixed integer programming, and domain-independent dynamic programming.
 468 We compared the performance of the models on a set of randomly generated instances. MIP
 469 and CP models were solved with commercial solvers, the DIDP model was solved using the
 470 recently developed domain-independent dynamic programming solver [15].

471 Our results show that domain-independent dynamic programming models are the fastest
 472 in both finding high-quality feasible solutions to problem instances and in solving them to
 473 optimality. For large instances, when the number of requests is greater than 15, even DIDP
 474 models were not able to solve the instances to the optimality. Although in general, MIP
 475 models were faster to find feasible solutions than CP, for short run times, CP found better
 476 solutions than both of the MIP models.

477 Our future work will study generalizations of the problem by considering multiple aircraft
 478 and more realistic representation of cargo size and aircraft capacity. We have also embarked
 479 on a study of the decomposition of the problem both to better fit the real-world use case where
 480 routes are often predefined and to exploit the computational advances of the mathematical
 481 structure of the decomposition.

482 **References**

-
- 483 **1** Maria Battarra, Jean-François Cordeau, and Manuel Iori. Chapter 6: pickup-and-delivery
 484 problems for goods transportation. In *Vehicle Routing: Problems, Methods, and Applications*,
 485 *Second Edition*, pages 161–191. SIAM, 2014.
- 486 **2** J. Benders. Partitioning procedures for solving mixed-variables programming problems.
 487 *Numerische mathematik*, 4(1):238–252, 1962.
- 488 **3** Sanjeeb Dash, Oktay Günlük, Andrea Lodi, and Andrea Tramontani. A time bucket formulation
 489 for the traveling salesman problem with time windows. *INFORMS Journal on Computing*,
 490 24(1):132–147, 2012.
- 491 **4** Irina Dumitrescu, Stefan Ropke, Jean-François Cordeau, and Gilbert Laporte. The traveling
 492 salesman problem with pickup and delivery: polyhedral results and a branch-and-cut algorithm.
 493 *Mathematical Programming*, 121:269–305, 2010.
- 494 **5** Pablo Factorovich, Isabel Méndez-Díaz, and Paula Zabala. Pickup and delivery problem with
 495 incompatibility constraints. *Computers & Operations Research*, 113:104805, 2020.
- 496 **6** M. S. Fox, N. Sadeh, and C. Baykan. Constrained heuristic search. In *Proceedings of the*
 497 *Eleventh International Joint Conference on Artificial Intelligence (IJCAI-89)*, pages 309–316,
 498 1989.
- 499 **7** Maria Gabriela S Furtado, Pedro Munari, and Reinaldo Morabito. Pickup and delivery
 500 problem with time windows: a new compact two-index formulation. *Operations Research*
 501 *Letters*, 45(4):334–341, 2017.

- 502 8 Jonas Hatzenbühler, Erik Jenelius, Győző Gidófalvi, and Oded Cats. Multi-purpose
503 pickup and delivery problem for combined passenger and freight transport. *arXiv preprint*
504 *arXiv:2210.05700*, 2022.
- 505 9 Tino Henke, M Grazia Speranza, and Gerhard Wäscher. The multi-compartment vehicle
506 routing problem with flexible compartment sizes. *European Journal of Operational Research*,
507 246(3):730–743, 2015.
- 508 10 Sin C Ho, Wai Yuen Szeto, Yong-Hong Kuo, Janny MY Leung, Matthew Petering, and
509 Terence WH Tou. A survey of dial-a-ride problems: Literature review and recent developments.
510 *Transportation Research Part B: Methodological*, 111:395–421, 2018.
- 511 11 Alexander Hübner and Manuel Ostermeier. A multi-compartment vehicle routing problem
512 with loading and unloading costs. *Transportation Science*, 53(1):282–300, 2019.
- 513 12 Serdar Kadioglu, Yuri Malitsky, Ashish Sabharwal, Horst Samulowitz, and Meinolf Sellmann.
514 Algorithm selection and scheduling. In *International conference on principles and practice of*
515 *constraint programming*, pages 454–469. Springer, 2011.
- 516 13 Richard M. Karp. Reducibility among combinatorial problems. In Raymond E. Miller and
517 James W. Thatcher, editors, *Proceedings of a symposium on the Complexity of Computer*
518 *Computations, held March 20-22, 1972, at the IBM Thomas J. Watson Research Center,*
519 *Yorktown Heights, New York, USA*, The IBM Research Symposia Series, pages 85–103. Plenum
520 Press, New York, 1972.
- 521 14 Ryo Kuroiwa and J. Christopher Beck. Domain-independent dynamic programming: Generic
522 state space search for combinatorial optimization. *Proceedings of the International Conference*
523 *on Automated Planning and Scheduling*, 33(1):236–244, Jul. 2023.
- 524 15 Ryo Kuroiwa and J. Christopher Beck. Solving domain-independent dynamic programming
525 problems with anytime heuristic search. *Proceedings of the International Conference on*
526 *Automated Planning and Scheduling*, 33:245–253, Jul. 2023.
- 527 16 Jin Li, Qihui Lu, and Peihua Fu. Carbon footprint management of road freight transport
528 under the carbon emission trading mechanism. *Mathematical Problems in Engineering*, 2015,
529 2015.
- 530 17 Clair E Miller, Albert W Tucker, and Richard A Zemlin. Integer programming formulation of
531 traveling salesman problems. *Journal of the ACM (JACM)*, 7(4):326–329, 1960.
- 532 18 Andrea Mor and Maria Grazia Speranza. Vehicle routing problems over time: a survey. *Annals*
533 *of Operations Research*, 314(1):255–275, 2022.
- 534 19 Salma Naccache, Jean-François Côté, and Leandro C Coelho. The multi-pickup and delivery
535 problem with time windows. *European Journal of Operational Research*, 269(1):353–362, 2018.
- 536 20 Manuel Ostermeier, Tino Henke, Alexander Hübner, and Gerhard Wäscher. Multi-compartment
537 vehicle routing problems: State-of-the-art, modeling framework and future directions. *European*
538 *Journal of Operational Research*, 292(3):799–817, 2021.
- 539 21 Sophie N Parragh, Karl F Doerner, and Richard F Hartl. A survey on pickup and deli-
540 very models part ii: Transportation between pickup and delivery locations. *Journal für*
541 *Betriebswirtschaft*, 58:81–117, 2006.
- 542 22 Yuan Qu and Jonathan F Bard. The heterogeneous pickup and delivery problem with
543 configurable vehicle capacity. *Transportation Research Part C: Emerging Technologies*, 32:1–20,
544 2013.
- 545 23 Oscar Tellez, Samuel Vercraene, Fabien Lehuédé, Olivier Péton, and Thibaud Monteiro. The
546 fleet size and mix dial-a-ride problem with reconfigurable vehicle capacity. *Transportation*
547 *Research Part C: Emerging Technologies*, 91:99–123, 2018.
- 548 24 Marjolein Veenstra, Kees Jan Roodbergen, Iris FA Vis, and Leandro C Coelho. The pickup
549 and delivery traveling salesman problem with handling costs. *European Journal of Operational*
550 *Research*, 257(1):118–132, 2017.
- 551 25 Weixiong Zhang. Complete anytime beam search. In *AAAI/IAAI*, pages 425–430, 1998.