Model Formulations for Pickup and Delivery Problems in Designated Driver Services

Alp Arslan
Department of Management Science, Lancaster University Management School, United Kingdom,
a.arslan1@lancaster.ac.uk

Niels Agatz
Rotterdam School of Management, Erasmus University Rotterdam, the Netherlands nagatz@rsm.nl

F. Jordan Srour
Department of IT and Operations Management, Lebanese American University, Lebanon
jordan.srour@lau.edu.lb

Designated driver services use company vehicles to deliver drivers to customers. The drivers then drive the customers from their origins to their destinations in the customers' own cars; at the destinations the drivers are picked up by a company vehicle. We typically see teams of drivers assigned to company vehicles serving customers. When, however, the drivers may be dropped off by one vehicle and picked up by another, a challenging, novel pick-up and delivery problem arises. In this paper, we introduce two formulations to solve this problem to optimality using a general purpose solver. In particular, we present a three-index and a two-index mixed integer program formulation to generate optimal, least-cost routes for the company vehicles and drivers. Using these MIPs, we find that the two-index formulation outperforms the three-index formulations by solving more instances to optimality within a given run time limit. Our computational experiments also show that up to 60% cost savings are possible from using a flexible operating strategy as compared to a strategy in which drivers and company vehicles stay together throughout a shift.

Key words: routing, dial-a-ride, pickup and delivery problem, designated driver services, routing with precedence

---

Drinking and driving is a problem in nearly every country. According to the National Highway Transportation Safety Administration in the United States of America, 10,598 people were killed in alcohol impaired driving crashes in 2020 (NHTSA 2020). One of the primary contributors to this problem is the desire of those who have been drinking to have their car at home the next morning. To combat this issue many charitable organizations and businesses have built a system whereby the person who has been drinking can call for a chauffeur who drives them home in their own car.

For example, Operation Red Nose in Quebec, Canada uses volunteers to drive inebriated persons home in their own cars; the drivers accept donations for this service that are subsequently donated to charities. Ride-share services such as Didi has been

offering designated driver services since 2015 (Horwitz 2015). Several start-ups in the U.S. and Australia offer designated driver services (Editors 2014, Fowler 2015) while in South Korea (Sang-Hun 2007) these services are long-standing and quite common. In the Netherlands, companies such as Beter Bob, Rent-a-Bob, and Super Bob fill this role. (Bob in the Benelux region is a slang term for designated driver.) These services typically use dedicated company cars to move the drivers between customer locations. In addition to the designated driver services, such companies also run day-time operations targeting business people and people who are unable to drive their own car home after a medical procedure.

The main advantage of using a designated driver service as compared to using a regular taxi, ride-hailing, or public transit service is that it allows people to immediately take their car back home with them. A designated driver service eliminates the inconvenience, costs and emissions of an additional back and forth trip to pick up their car later. Moreover, in some urban areas, it may not be possible to leave the car behind due to strict overnight parking regulations.

The service of moving cars with a team of drivers is not unique to the designated driver business. For example, one-way car rental services such as Car2Go regularly need to re-balance the vehicles in their network when they accumulate at popular destinations while becoming depleted at popular origins (Nourinejad et al. 2015). In a similar vein, car dealerships that sell cars online can use a team of drivers to deliver the cars to their customers. Luxury car brands such as Lexus offer a service to pickup the customers' cars at their homes when needing to go to the garage for maintenance or repairs.

Whether driving a customer's car for maintenance or a customer in their car to prevent drunk driving, the general operations in a designated driver service are as follows: (i) a vehicle delivers a designated driver ("driver" for short) to the customer's origin, (ii) the driver drives the customer to their destination in the customer's car, (iii) a vehicle picks up the driver at the customer's destination. This gives rise to the optimization problem of determining how to serve all customer requests with a given fleet of drivers and vehicles. We refer to this problem as the *Designated Driver Problem (DDP)*

The DDP falls within the realm of Vehicle Routing Problems with Pickup and Delivery, in particular, the Pickup and Delivery Problem with Time Windows (PDPTW). However, the DDP is actually a 'delivery and pickup' problem. A driver delivered by

one vehicle may be subsequently picked up by another. This creates the need to synchronize between different vehicle routes. Furthermore, drivers, once dropped off, move autonomously using customer cars. As the driver can only be picked-up at the customer's destination after arriving, the pickup time windows are 'dynamic' and endogenous (Gschwind et al. 2012). Despite the myriad of PDPTW formulations (Aziez et al. 2020, Furtado et al. 2017, Parragh et al. 2008, Berbeglia et al. 2007), these don't account for the specific complexities of the DDP. The synchronization between the different routes makes it impossible to define 'small' local search neighborhoods, e.g., moving a delivery task between two routes may impact a third route that handles the corresponding pickup task. Therefore, it is unclear how to effectively formulate a model to solve this problem with a general-purpose solver.

The contributions of this paper are as follows. First, we formalize the designated driver problem – a problem variant in the family of pickup and delivery problems. Second, we propose two integer programming formulations to solve the designated driver problem using a general-purpose solver. Specifically, we develop a tailored two-index formulation to overcome the computational limitations of a simpler three-index formulation. Third, we present a computational study to compare the performance of our proposed formulations and provide insights into the advantages and disadvantages of different operating strategies for designated driver services. Our results show that the two-index formulation significantly outperforms the three-index formulation. We also show that we can solve realistic-sized instances with flexible routing strategies to optimality using the two-index formulation. The speed and optimality of these routing solutions may directly benefit practitioners running designated driver services. Moreover, we believe that our two-index formulation can be a valuable starting point for further research in this area.

The remainder of the paper is organized as follows. The next section provides an overview of the relevant literature. Section 2 gives a formal problem description, including the notation. Sections 3 and 4 present three and two-index mixed-integer programming formulations for the Designated Driver Problem, respectively. Section 5 describes our computational experiments and results. The paper concludes with a discussion and suggestions for future work in Section 6.

1. Related Literature

The Designated Driver Problem (DDP) is part of the general class of Vehicle Routing Problems termed Pickup and Delivery with Time Windows (PDPTW) that forms a subclass of the Capacitated Vehicle Routing Problem with Time Windows. For more comprehensive overviews of pickup and delivery problems, see Parragh et al. (2008), Berbeglia et al. (2007).

In the DDP, a generalization of the PDPTW, the company vehicle capacity can be greater than one *and* the customer origins (driver drop-offs) may be served by a different company vehicle than the customer destinations (driver pick-ups). This is at the heart of what differentiates the DDP from traditional pickup and delivery problems: the company vehicles do not directly transfer customers or goods between their origins and destinations but rather transport the drivers between these customer locations. This means that the exogenous customer requests do not directly correspond to the driver transportation requests. In particular, while the origin-destination pairs of the customer requests are given, we decide on the routing of the drivers, i.e., the origin-destination pairs of the drivers. This is similar to the VRP with divisible deliveries and pickups, however, the delivery load and pickup load (while associated with one client) are geographically disparate (Nagy et al. 2013). This in-turn gives rise to dependencies between the customer time windows for pickup (driver drop-off) and the driver time windows for pick-up (customer drop-off).

Using the terminology of Berbeglia et al. (2007), from the perspective of the drivers, the DDP can be classified as a "one-to-one" pickup and delivery problem in which each transportation request has one origin and one destination. From the perspective of the company vehicles, however, the DDP becomes a vehicle routing problem with precedence constraints. The literature on both strategies is reviewed in turn here.

In a review of models and algorithms for one-to-one pickup and delivery problems, Cordeau et al. (2008) present a two-index formulation for the single vehicle pickup and delivery problem, yet when moving to the multi-vehicle pickup and delivery problem they introduce a three-index model. Interestingly Cordeau et al. (2008) note in the same review article that when considering the PDPTW as a dial-a-ride problem (DARP) Ropke and Cordeau (2009) find the superiority of the two-index formulation. In terms of the DDP, the time dependencies between driver drop-off and driver pick-up are similar to those arising in DARPs (Gschwind et al. 2012).

Similar to the DARPs, the DDP focusses on the pickup and delivery of people. The temporal dependencies in the DDP are conceptually similar to the ride-time constraints of DARPs. That is, a driver can only be picked up at the customer's destination after he/she has driven the customer to this destination. As such, the earliest pickup time at the customer destination depends on the time that the service starts at the customer's origin and this depends on the time that the driver is dropped-off at the customer's origin. Moreover, since we specify a maximum waiting time for the driver at the destination, the problem also involves 'dynamic' time windows on the driver pickup (Gschwind et al. 2012).

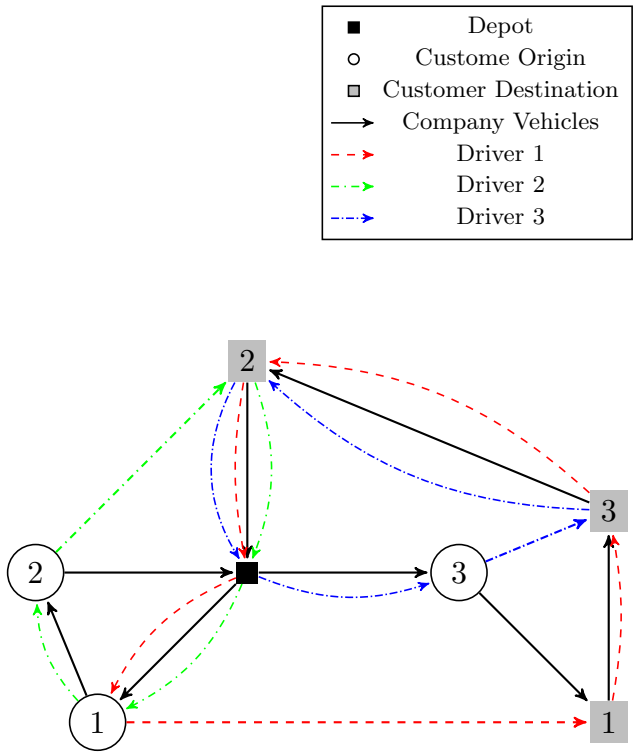
The requirement that the driver drop off occurs before the driver pickup also brings the DDP into the realm of the Pick-up and Delivery Traveling Salesman Problem with Precedence Relationships (Gouveia and Ruthmair 2015). A problem that Gouveia and Ruthmair (2015) documented well in both a single and multi-commodity form using a two-index formulation. While their formulation must respect time-based dependencies arising from the precedence, they do not include time windows for the initial stop at the job's origin.

In addition to the dependencies created in the time windows, allowing drivers to be picked up by a different vehicle than the one involved in the drop-off gives rise to temporal dependencies between different routes. Figure 1 illustrates a case in which three customers are served by two company vehicles and three drivers. In this illustration, one vehicle leaves the depot with two drivers, but returns with none; in contrast, the second vehicle leaves the depot with one driver but returns with three.

This interplay of routes links the DDP to the stream of literature on vehicle routing problems with route synchronization (Drexl 2012). The synchronization of different vehicles that jointly serve a customer request also arises in pickup and delivery problems with transfers or transshipments that allow for the possibility of transferring passengers or items between different vehicles at pre-defined transfer locations see e.g., (Danloup et al. 2018, Maknoon and Laporte 2017, Rais et al. 2014, Masson et al. 2014, 2012, Cortés et al. 2010). Across this genre of pick-up and delivery problem, the preference appears to be for three-index formulations.

Another domain in which these route dependencies arise is that of drone delivery applications that combine both regular and autonomous vehicle movements to serve demands (Yu et al. 2022, Macrina et al. 2020, Chung et al. 2020, Karak and Abdelghany

Figure 1 **An example DDP solution with flexible pairing between company vehicles and chauffeurs.**



2019, Otto et al. 2018). In settings with a single drone and a single truck one can formulate the problem using 'operations' as the building blocks to capture all activities when the drone and truck are separated (Agatz et al. 2018). This involves first enumerating all possible operations. This, however, cannot be done in settings with multiple drones per vehicle or flexible drone and truck assignments (Boysen et al. 2018). Again, in this genre of pick-up and delivery problem, formulations typically use three or more indices.

Existing studies show mixed results on the computational performance of two-index and three-index formulations for pick-up and delivery problems. Furtado et al. (2017) show that the two-index formulation computationally outperforms the three-index formulation for the PDPTW. The paper by Aziez et al. (2020) compares a two-index and three-index formulation, and asymmetric representation (AR), for multi-depot pick-up and delivery problems with time windows. Their computational study shows that the AR and three-index formulation outperform the two-index formulation. As such, it is not clear which formulation is best suited for our specific problem. We contribute to this line of research by introducing both a three-index and two-index formulation for the DDP. We demonstrate the capabilities of these formulations relative to the two operating strategies of fixed and flexible teams.

2. Problem Description

In this paper, we focus on a static problem setting in which all requests are known before planning. The static model of the DDP is relevant to settings in which customer requests are placed in advance, e.g., customer requests are placed during the daytime for service later that evening. Moreover, the static problem provides a natural starting point to study dynamic settings within a rolling horizon framework.

We consider a designated driver service that needs to serve a set $\mathcal{R}$ of customer requests using a team of B drivers and a fleet of homogeneous company vehicles $\mathcal{V}$. Each company vehicle $v \in \mathcal{V}$ starts and ends at the depot and can carry at most Q drivers. Note that company vehicles always have a company driver who remains with the vehicle and is not included in the capacity, Q.

Let $\mathcal{P}$ represent all origins of the customer requests in $\mathcal{R}$ and $\mathcal{D}$ represent all destinations of the customer requests in $\mathcal{R}$. Each customer request $r \in \mathcal{R}$ requires a driver to transport them from a customer origin $o_r \in \mathcal{P}$ to a destination (driver pick-up location) $p_r \in \mathcal{D}$. Each request $r \in \mathcal{R}$ has a time window $[e_r, l_r]$ with an earliest e_r and latest time l_r that service can begin at o_r. The time windows reflect the fact that customers typically allow some flexibility around their desired pickup time.

The objective is to find tours for a set of capacity constrained company vehicles and drivers that minimize the costs of serving customer requests plus the penalty costs associated with rejecting a customer request.

In practice, to simplify planning the routes for the company vehicles and drivers, companies often use *fixed pairs* of a company vehicle and a single driver. This means that each driver works with one particular company vehicle. With this operational structure, the problem can be modeled as a truckload pickup and delivery problem (Srour et al. 2018). Extending beyond this capacity constrained cases, we examine a capacity of up to three drivers per vehicle in both a *fixed* and *flexible* mode of operations. In the *flexible* mode of operations, there is no restriction on pairing between drivers and company vehicles; drivers can be dropped-off by one company vehicle and picked up by another. While this operational structure makes planning more complex, it allows for more routing flexibility.

In the flexible mode of operation, while the total number of company vehicles and drivers is set a priori, there is flexibility in the specific assignment of drivers to company vehicles when departing from the depot. There is also flexibility in routing as a driver

who is dropped off at origin o_r by one vehicle can be picked up at destination p_r by the same vehicle or another vehicle.

In the flexible pairs case, it might be necessary to drop off a driver earlier than the start of the customer time window or pick up the driver beyond the end of service. The time that a driver waits to begin service is only considered waiting when the driver is alone; there is no restriction on waiting time when the driver is within the company vehicle (as in the fixed pairs case). To handle waiting time, we specify a maximum waiting time at the customer location for the driver. In particular, a driver can wait at most W_{o_r} between being dropped off and starting service at the customer's origin. Similarly, a driver can wait at most W_{p_r} between arriving at a customer's destination and being picked up. In practice, the specific value of this maximum waiting time may depend on various factors such as weather conditions and whether or not it is possible to wait inside. That is, the driver may be willing to wait longer inside a nearby restaurant than outside on the street.

In this context, we study different model formulations across four operating strategies as summarized in Table 1.

Table 1 Summary of models tested across the specified operating strategies.

	3-index	2-index
Fixed	Section 3	Section 4
Flexible	Section 3 without constraint (5)	Section 4 without constraints (24) - (28)

As can be seen in this table, all three-subscript models are derived from the fixed team three-subscript formulation just as all two-subscript models are derived from the fixed team two-subscript formulation. To that end, the following two subsections present those formulations in detail.

3. Three-index Formulation

3.1. Decision Variables

This model version follows a traditional three subscript structure on a graph $G = (\mathcal{N}, E)$, where $\mathcal{N}$ is the union of nodes representing the locations associated with the requests $\mathcal{P} \cup D$, where $\mathcal{P}$ represents the customer pick-up locations and $\mathcal{D}$ represents the customer drop-off locations. To model the fixed pairing, we use a three-subscript formulation in which the binary decision variable, x_{ij}^v, indicates whether arc (i, j) is included in the route of vehicle v or not. The associated c_{ij} represent the cost of traveling

on arc (i, j). The variable y_r, indicates whether request r is served (1) or not (0) and c_r represent the rejection costs.

In addition to the route-related variables, we also have time-related variables to ensure that time windows are obeyed. Let $a_i^v \in \mathbb{R}^+$ be the arrival time of vehicle v at node i and $d_i^v \in \mathbb{R}^+$ the departure time of vehicle v from node i. Furthermore, to restrict the waiting times of the drivers at the origin and the destination of each request, we need to model the time at which the drivers start their service. Let $s_r \in \mathbb{R}^+$ be the start of service from the origin o_r of request r. At this time, a driver starts driving from o_r to the request's destination p_r; a distance that requires t_r time. Thus, the arrival time of the driver at the request's destination p_r is given by $s_r + t_r$.

Finally, to obey the capacity constraint of vehicle v we let q_i^v represent the number of drivers in the company vehicle when departing from node i.

For ease of reference, Table 2 presents the notation used in the three-subscript formulation of the DDP.

Table 2 Summary of notation used in the three-index formulation

x_{ij}^v	Binary decision variable indicating if vehicle v traverses arc (i, j) or not.
y_r	Binary decision variable indicating if request r is served (1) or not (0).
a_i^v	Arrival time of vehicle v at node $i \in \mathcal{P} \cup D$.
d_i^v	Departure time of vehicle v from node $i \in \mathcal{P} \cup D$.
s_r	The start time for service from o_r of request r.
q_i^v	The number of drivers in vehicle v when departing node $i \in \mathcal{N}$.

3.2. The Model

The objective (1) minimizes the sum of the cost of serving or rejecting the requests.

$$\min \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}} c_{ij} x_{ij}^v + \sum_{r \in \mathcal{R}} c_r (1 - y_r) \tag{1}$$

Constraint sets (2) - (4) ensure that each node is visited at most once and that the request origin needs to be visited if the request is served. Constraints (5) make sure that the same vehicle visits both the origin and the destination of a request. We don't need this constraint in the flexible strategy.

$$\sum_{v \in \mathcal{V}} \sum_{i \in \mathcal{N}} x_{ij}^v \leq 1, \quad j \in \mathcal{N} \tag{2}$$

$$\sum_{v\in\mathcal{V}}\sum_{j\in\mathcal{N}} x^v_{ij} \leq 1, \quad i \in \mathcal{N} \tag{3}$$

$$\sum_{i\in\mathcal{N}}\sum_{v\in\mathcal{V}} x^v_{io_r} = y_r \quad r \in \mathcal{R} \tag{4}$$

$$\sum_{i\in\mathcal{N}} x^v_{io_r} - \sum_{j\in\mathcal{N}} x^v_{jp_r} = 0 \quad v \in \mathcal{V},\ r \in \mathcal{R} \tag{5}$$

Constraints (6) - (9) guarantee that the time that the company vehicle arrives at each location is time feasible. Constraint set (6) ensures that the company vehicle arrives to a node before it departs from that node. Constraint set (7) ensures that the arrival to a request origin occurs before the arrival to a destination in order to ensure that any customer pick-ups on the route occur before any customer drop-offs. Constraint sets (8) and (9) ensure that a company vehicle arrives to a subsequent node only after it has departed from the previous node and traveled the required amount of time.

$$a^v_j \leq d^v_j \quad j \in \mathcal{P}\cup D,\ v \in \mathcal{V} \tag{6}$$

$$a^v_{p_r} \geq a^v_{o_r}, \quad r \in \mathcal{R}\ v \in \mathcal{V} \tag{7}$$

$$a^v_j \geq d^v_i + t_{ij} - M(1-x^v_{ij}), \quad i,j \in \mathcal{P}\cup D, v \in \mathcal{V} \tag{8}$$

$$a_j \leq d_i + t_{ij} + M(1-x_{ij}), \quad i,j \in \mathcal{P}\cup D \tag{9}$$

To restrict the waiting times of the drivers at the origin and the destination of each request, we need to model the time at which the drivers start their service.

Constraints (10) - (12) restrict the start time of the service at the origin of a request. Constraint set (10) ensures that service cannot start before a driver has arrived and constraint set (11) makes sure that a driver does not wait alone for longer than allowed before starting service. Constraints (12) make sure that the driver starts service within the service time window. The pickup times of the driver at the destination p_r are assigned according to constraints (13) and (14). Constraint set (13) makes sure that the company car picks up the driver before his/her maximum waiting time. Constraint

set (14) ensures that the company vehicle does not depart the customer drop-off (driver pickup) location before the driver has arrived. One can set M to the length of service duration.

$$s_r \geq a^v_{o_r} - M(1 - x^v_{io_r}), \quad r \in \mathcal{R}, \quad i \in \mathcal{N} \quad v \in \mathcal{V} \tag{10}$$

$$s_r \leq d^v_{o_r} + W_{o_r} + M(1 - x^v_{io_r}), \quad r \in \mathcal{R}, \quad i \in \mathcal{N} \quad v \in \mathcal{V} \tag{11}$$

$$e_r \leq s_r \leq l_r, \quad r \in \mathcal{R} \tag{12}$$

$$a^v_{p_r} \leq s_r + t_r + W_{p_r} + M(1 - x^v_{p_r j}), \quad r \in \mathcal{R}, \quad j \in \mathcal{N}, \quad v \in \mathcal{V} \tag{13}$$

$$s_r + t_r \leq d^v_{p_r} + M(1 - x^v_{p_r j}), \quad r \in \mathcal{R}, \quad j \in \mathcal{N}, \quad v \in \mathcal{V} \tag{14}$$

We define the number of drivers in company vehicle v at the departure of node i by q^v_i. Constraints (15) and (16) enforce load balance in terms of the number of drivers in the company vehicles. In particular, these constraints ensure that whenever the arc (i, j) is traveled, the number of drivers in the company car is decreased or increased by 1, depending on whether node j is a customer origin (driver drop-off) or a customer destination (driver pickup), respectively. We do not specify which driver is dropped off but make sure that there is at least one driver available in the company car when visiting an origin. In order to keep track of the number of drivers in the company cars, we make use of a data vector A, where A_i is equal to -1 if node i is a request origin and 1 if i is a request destination, i is in the set $\mathcal{P} \cup D$.

Constraint set (17) limits the number of drivers in a company vehicle at any point in time to the maximum capacity Q, while constraint set (18) guarantees that the total number of drivers leaving the depot does not exceed B. The load balance constraints along with the flow constraints ensure that all drivers who leave the depot also return to the depot.

$$q^v_j \leq q^v_i + A_j + Q(1 - x^v_{ij}), \quad i \in \mathcal{N}, j \in \mathcal{P} \cup D, v \in \mathcal{V}, i \neq j \tag{15}$$

$$q_j \geq q_i + A_j - Q(1 - x_{ij}), \quad i \in \mathcal{N}, j \in \mathcal{P} \cup D, i \neq j \tag{16}$$

$$q_i^v \leq Q, \quad i \in \mathcal{N} \tag{17}$$

$$\sum_{v \in \mathcal{V}} q_0^v \leq B \tag{18}$$

4. Two-index Formulation

This formulation is based on the two-index formulation of Srour et al. (2018) in which each driver remains paired with a company vehicle throughout the operations and Furtado et al. (2017) in which the pick-up and delivery functions occur on a single vehicle's route but may be separated by intervening pick-ups or deliveries.

4.1. Decision variables and parameters

In this formulation, we exploit a two-index formulation in which the binary decision variable, x_{ij}, indicates whether arc (i, j) is included in the route or not. If i is a node in $\mathcal{V}$ and j is a node in $\mathcal{P} \cup D$, then arc (i, j) represents vehicle i serving location j first from the depot. Similarly, if i is in $\mathcal{P} \cup D$ and j is in $\mathcal{V}$, then arc (i, j) represents location i as the last on the route before returning to the depot. The associated c_{ij} represents the cost of traveling to or from the depot. When both i and j are in $\mathcal{V}$ then arc (i, j) corresponds to vehicle i remaining idle at the depot. If both i and j are nodes in $\mathcal{P} \cup D$, then (i, j) represents location j following location i on a route where c_{ij} is the cost of traveling on arc (i, j). If both i and j are in $\mathcal{P} \cup D$ and $i = j$, then (i, j) is associated with the rejection of request i. In order to track these rejections, we introduce a decision variable y_r which takes a value of 1 if job r is served and 0 if it is not served; c_r is the cost associated with rejecting request r.

Decision variable v_j specifies the route number that job $j \in \mathcal{P}$ is on; in this way, we can ensure that the origin and destinations of each job end up on the same route. Through this convention, we can keep the driver teams paired with the company vehicles.

In addition to the route-related variables, we also have time-related variables to ensure that time windows are obeyed. Let $a_i \in \mathbb{R}^+$ be the arrival time at node i and $d_i \in \mathbb{R}^+$ the departure time of a company vehicle from node i. Furthermore, to restrict the waiting times of the drivers at the origin and the destination of each request, we need to model the time at which the drivers start their service. Let $s_r \in \mathbb{R}^+$ be the start

of service from the origin o_r of request r. At this time, a driver starts driving from o_r to the request's destination p_r; a distance that requires t_r time. Thus, the arrival time of the driver at the request's destination p_r is given by $s_r + t_r$.

Finally, to obey the capacity constraints of the company vehicles we let q_i represent the number of drivers in the company vehicle when departing from node i.

For ease of reference, Table 3 provides a summary of the notation used in the two-subscript formulation.

Table 3 Summary of notation used in the two-index formulation.

x_{ij}	Binary decision variable indicating if arc (i,j) is included on the route or not.
y_r	Binary decision variable indicating if request r is served (1) or not (0).
v_j	Integer decision variable specifying the vehicle route on which job $j \in \mathcal{P} \cup D$ is served.
a_i	Arrival time at node $i \in \mathcal{P} \cup D$.
d_i	Departure time from node $i \in \mathcal{P} \cup D$.
s_r	The start time for service from o_r of request r.
q_i	The number of drivers in the company car when departing node $i \in \mathcal{N}$.

4.2. The Model

The objective (19) minimizes the sum of the cost of serving or rejecting the requests.

$$\min \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}} c_{ij} x_{ij} + \sum_{r \in \mathcal{R}} c_r (1 - y_r) \tag{19}$$

Constraint sets (20) - (23) ensure that each node is included in exactly one tour and that if the origin node is not visited, the destination node is similarly not visited – representing a job rejection.

$$\sum_{i \in \mathcal{N}} x_{ij} = 1, \quad j \in \mathcal{N} \tag{20}$$

$$\sum_{j \in \mathcal{N}} x_{ij} = 1, \quad i \in \mathcal{N} \tag{21}$$

$$\sum_{i \in \mathcal{N}/o_r} x_{io_r} = y_r \quad r \in \mathcal{R} \tag{22}$$

$$x_{p_r p_r} = (1 - y_r) \quad r \in \mathcal{R} \tag{23}$$

Constraint sets (24) - (28) serve to make sure that the same drivers remain paired to the company cars or rather that the drop-off of a particular driver occurs on the same route as the pick-up of a particular driver. We don't need these constraint in the flexible strategy. Specifically, (24) and (25) force the value of v_j to j if j is the first node on a route; otherwise v_j may take on any value between 0 and V, the number of vehicles. Constraint sets (26) and (27) ensure that all jobs on the same route are assigned the same value for v_j. Finally, constraints (28) ensure that the customer pick-up for a request occurs on the same route as the customer drop-off. The value of M can be set to the number of requests for constraints (25)- (28).

$$v_j \geq j x_{ij} \quad i \in \mathcal{V}; j \in \mathcal{P} \tag{24}$$

$$v_j \leq j x_{ij} + M(1 - x_{ij}) \quad i \in \mathcal{V}; j \in \mathcal{P} \tag{25}$$

$$v_j \geq v_i - M(1 - x_{ij}) \quad i, j \in \mathcal{P} \cup D \tag{26}$$

$$v_j \leq v_i + M(1 - x_{ij}) \quad i, j \in \mathcal{P} \cup D \tag{27}$$

$$v_{R+i} = v_i \quad i \in \mathcal{P} \tag{28}$$

Constraints (29) - (30) guarantee that the time that the company vehicle arrives at each location is time feasible. Constraint set (29) ensures that the company vehicle arrives at a node before it departs from that node. Constraint set (30) ensures that a company vehicle arrives at a subsequent node only after it has departed from the previous node and traveled the required amount of time. The minimum value for M in Constraints 30 is the end of the service day.

$$a_j \leq d_j \quad j \in \mathcal{P} \tag{29}$$

$$a_j \geq d_i + t_{ij} - M(1 - x_{ij}), \quad i, j \in \mathcal{P} \cup D \tag{30}$$

Constraints (31)-(33) restrict the start time of the service at the origin of a request. Constraint set (31) ensures that service cannot start before a driver has arrived and constraint set (32) makes sure that a driver does not wait alone for longer than allowed before starting service. Constraints (33) make sure that the driver starts service within the service time window. The pick up times of the driver at the destination p_r are assigned according to constraints (34) and (35). Constraint set (34) makes sure that the company car picks up the driver before his/her maximum waiting time. Constraint set (35) ensures that the company vehicle does not depart the customer drop-off (driver pickup) location before the driver has arrived.

$$s_r \geq a_{o_r}, \quad r \in \mathcal{R} \tag{31}$$

$$s_r \leq d_{o_r} + W_{o_r}, \quad r \in \mathcal{R} \tag{32}$$

$$e_r \leq s_r \leq l_r, \quad r \in \mathcal{R} \tag{33}$$

$$a_{p_r} \leq s_r + t_r + W_{p_r}, \quad r \in \mathcal{R} \tag{34}$$

$$s_r + t_r \leq d_{p_r}, \quad r \in \mathcal{R} \tag{35}$$

We define the number of drivers in each company vehicle at the departure of node i by q_i. Constraints (36) and (37) enforce load balance in terms of the number of drivers in the company vehicles. In particular, it ensures that whenever the arc (i, j) is traveled, the number of drivers in the company car is decreased or increased by 1, depending on whether node j is a customer origin (driver drop-off) or a customer destination (driver pickup), respectively. In this stage, we do not specify which driver is dropped off but make sure that there is at least one driver available in the company car when visiting an origin. In order to keep track of the number of drivers in the company cars, we make use of a data vector A, where A_i is equal to -1 if node i is a request origin and 1 if i is a request destination, i is in the set $\mathcal{P}$.

Constraint set (38) limits the number of drivers in a company vehicle at any point in time to the maximum capacity Q, while constraint set (39) guarantees that the total

number of drivers leaving the depot does not exceed B. The load balance constraints along with the flow constraints ensure that all drivers who leave the depot also return to the depot.

$$q_j \leq q_i + A_j + Q(1 - x_{ij}), \quad i \in \mathcal{N}, j \in \mathcal{P}, i \neq j \tag{36}$$

$$q_j \geq q_i + A_j - Q(1 - x_{ij}), \quad i \in \mathcal{N}, j \in \mathcal{P}, i \neq j \tag{37}$$

$$\max\{0, A_i\} \leq q_i \leq \min\{Q, Q + A_i\}, \quad i \in \mathcal{N} \tag{38}$$

$$\sum_{i \in \mathcal{V}} q_i \leq B \tag{39}$$

5. Computational Study

In this section, we describe our computational study to test the three-index and two-index formulations across the fixed and flexible operating strategies. Section 5.1 presents the instances and parameter settings; Section 5.2 compares the run times of the different formulations; and Section 5.3 evaluates the operational impacts of the fixed and flexible team strategies.

5.1. Test instances and testing environment

In our experiments, we use a random selection of the BUS instances from Srour et al. (2018). We use three sets of 20 instances for a total of 60 instances based on operational data from a designated driver service company. Each set of 20 instances has a different number of requests: 10, 20, and 30 requests. Within these requests across these instance sets, three, six and nine requests, respectively, go from the center of a 100 by 100 square region to the outer areas of the region; two, four, and six requests, respectively, go from the outer areas to the center of the region; and five, ten, and fifteen requests, respectively, go between randomly selected origins and destinations in the region. The depot, which is the starting and ending point for the company vehicles and drivers, is located at the lower left corner of the region (point [0,0] in a Cartesian grid) in all instances.

We ran our experiments on a computer with a 2.4 Gigahertz Intel processor and 8 GB installed RAM. The models were implemented in C++ using GUROBI 9.5

(Gurobi Optimization 2022) as our IP solver setting a maximum run time of 1800 seconds (30 minutes) with default parameter settings. We use the fixed-team with the unit capacity solution as a warm start for all formulations.

Unless stated otherwise, we use the following default parameters. The opening of the time windows at the origin of the requests were as specified in the instances with time windows set to five minutes *i.e.*, $l_r - e_r = 5$. We allow drivers to wait at a customer location for at most five minutes, *i.e.*, $W_{o_r} = W_{p_r} = 5$. All instances had nine company vehicles and 15 driers available. We run all of the formulations through the instances sets with varying vehicle capacities of one, two, or three, *i.e.*, $Q = 1, 2, 3$.

5.2. Formulations' computational performances

To compare the performance of the three-index and two-index formulations, Table 4 reports the number of instances solved to optimality within 1800 seconds. Table 5 reports the median and maximum run time bounded by 1800 seconds; and Table 6 shows the median and max optimality gaps of the instances that could not be solved to optimality within 1800 seconds.

Table 4 Number of instances solved to optimality within 1800 seconds

		Three-index		**Two-index**	
Cust	Q	fixed	flexible	fixed	flexible
10	1	20	20	20	20
10	2	20	20	20	20
10	3	20	20	20	20
20	1	20	4	20	20
20	2	0	0	14	20
20	3	0	0	13	20
30	1	6	0	19	20
30	2	0	0	1	20
30	3	0	0	1	20
Total		86	64	128	180

The results in Table 4 show that the two-index formulation clearly outperforms the three-index formulation. Using the two-index formulation, we solve significantly more instances to optimality within the maximum run time. In particular, we can solve all 180 instances for the flexible team setting and 128 instances for the fixed team setting. On the other hand, the three-index formulation only solves 64 instances within the flexible team setting and 86 instances within the fixed team setting. Interestingly, the

Table 5 Run times, median (max) seconds, for instances solved to optimality within 1800 seconds

		Three-index		Two-index	
Cust	Q	Fixed	Flexible	Fixed	Flexible
10	1	4.2 (9.7)	23.4 (69.4)	0.4 (0.5)	0.1 (0.1)
10	2	27.1 (95.3)	27.1 (95.3)	0.3 (0.8)	0.1 (0.2)
10	3	32.3 (74.1)	32.3 (74.1)	0.4 (0.9)	0.1 (0.2)
20	1	144.7 (758.4)	845 (1480.5)	0.7 (3.7)	0.1 (0.2)
20	2	n/a	n/a	80.8 (696.5)	0.6 (1.0)
20	3	n/a	n/a	121.8 (613.7)	0.5 (1.4)
30	1	852.3 (1576.1)	n/a	76.2 (1298.9)	0.2 (0.2)
30	2	n/a	n/a	1295.1 (1295.1)	5.9 (16.6)
30	3	n/a	n/a	974.3 (974.3)	3.8(10.7)

three-index formulation performs relatively better on the fixed team instances while the two-index formulation excels on the flexible team cases.

We see a similar pattern when looking at the run times in Table 5. For the flexible team case, we see that the two-index formulation solves all instances in less than 17 seconds. This indicates that we can use an exact solver in all practical settings without the need to rely on a heuristic approach.

Table 6 Percentage optimality gaps (median(max)) for instances solved time bound of 1800 seconds

		Three-index		Two-index	
Cust	Q	fixed	flexible	fixed	flexible
20	1	0	6.4 (10.4)	0	0
20	2	30.1 (42.4)	31.2 (37.8)	5.7 (31.5)	0
20	3	35.7 (50.2)	29.9 (34.8)	6.1 (28.3)	0
30	1	5.1 (17.2)	5.4 (45.9)	2.9 (2.9)	0
30	2	53.7 (69.9)	45.1 (53.1)	26.0 (45.6)	0
30	3	65.1 (74.0)	43.7 (53.2)	28.1 (47.5)	0

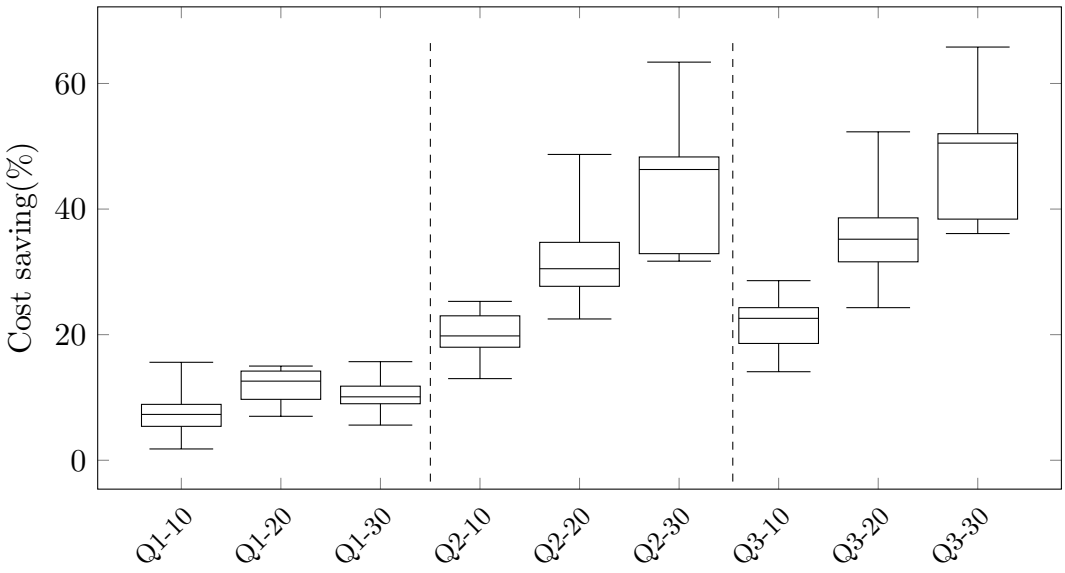
Table 6 reports the median and maximum optimality gaps, *i.e.*, $(UB - LB)/UB$, for the instances that did not achieve a provably optimal solution within 1800 seconds. These results show that for these instances, the two-index formulation results in smaller gaps than the three-index formulation. This again suggests that the two-index formulation is computationally superior.

5.3. Operational performance

This section presents the operational benefits of flexible teams as compared to fixed teams with varying vehicle capacities across the three instance sets. We use our best performing formulation, i.e., the 2-index formulation, to derive the savings. Figure 2 provides the proportional total cost savings if the flexible team strategy is chosen

instead of the fixed team strategy per instance. The cost savings are calculated by taking $\frac{z_{fixed}-z_{flex}}{z_{flex}}$, where z_{fixed} and z_{flex} denote the total costs for the fixed and flexible operating strategies, respectively.

Figure 2 Cost saving of flexible team as compared to fixed team, W=5, n=20



The results in Figure 2 show savings of up to 60% for the largest instances. We observe that the savings increase with the size of the instances, particularly when the capacity of the company vehicle is larger. We also see significant benefits of flexibility in the cases with more than unit capacity, i.e., $Q = 2$ and $Q = 3$. One potential reason is that the flexible strategy can exploit the additional wiggle room in capacity. It is, for example, easier to let drivers move around between vehicles.

Table 7 reports the *cost per served request* and the number of *rejected requests* for different instance sizes and vehicle capacities. We see that the costs per request decrease with the number of customers for the flexible strategy but not for the fixed strategy. This suggests that the flexible strategy is capable of using the additional routing flexibility to reduce vehicle miles. We see that a larger vehicle capacity reduces the cost per request for both strategies.

Table 8 reports the number of company vehicles and drivers across solutions. The maximum number of vehicles is nine and the maximum number of drivers is 15. As expected, we use more vehicles and drivers when serving more requests. The solutions with larger vehicle capacities use fewer vehicles and/ or more drivers. In some instances,

Table 7 **Average cost per served order and rejected requests, W=5, n=20**

		Cost per served request		Rejected	
Cust	Q	fixed	flexible	fixed	flexible
10	1	119.8	112.3	0	0
10	2	118.9	97.8	0	0
10	3	118.9	96.5	0	0
20	1	125.4	112.0	0.05	0.05
20	2	120.4	82.4	0.05	0
20	3	120.4	77.8	0.05	0
30	1	146.5	131.8	1.15	1.15
30	2	126.9	70.8	0.7	0
30	3	126.7	65.8	0.7	0

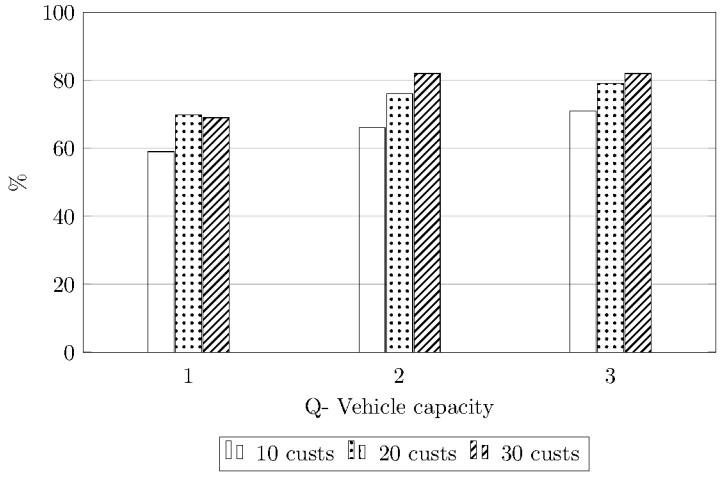
it is possible to serve more customers by deploying more drivers. In other instances, we serve the same number of customers with the same number of drivers but fewer vehicles. Note that in the flexible Q=1 case, we sometimes use more vehicles than drivers. The addition of an 'empty' vehicle adds slack capacity which allows for more flexibility in connecting different pickups and deliveries.

Table 8 **Average number of deployed company vehicles and drivers,** $W = 5$, $n = 20$

		Fixed			Flexible		
	Cust	Q=1	Q=2	Q=3	Q=1	Q=2	Q=3
No. of vehicles	10	4.1	4.1	4.1	4.5	3.9	3.9
	20	8.1	7.7	7.7	8.7	6.9	6.8
	30	9.0	9.0	9.0	9.0	8.3	7.9
No. of drivers	10	4.1	4.2	4.2	4.1	4.8	5.6
	20	8.1	8.7	8.7	8.0	9.5	10.8
	30	9.0	10.8	10.8	9.0	12.1	14.3

One prominent feature of the flexible operating strategy is that a driver can be dropped off by one company vehicle and picked up by another vehicle. We call this phenomenon a *swap*. Figure 3 reports the average percentage of swaps across all instances. This is the number of times a driver switches between vehicles divided by the total number of requests served. We see that the number of swaps increases with the number of customers and with the company vehicle's capacity. We also see that it is these swaps, emanating from flexibility, that are likely driving the large jump in cost savings seen between $Q = 1$ and $Q = 2$.

Figure 3 **Percentage of requests served by two different vehicles**



6. Concluding remarks and outlook

This paper formalizes the designated driver problem which arises in services where drivers transfer customers from their origins to their destinations using the customers' cars. In formalizing this challenging pick up and delivery problem, we are able to illustrate the importance of model-selection in drawing conclusions about solvability. Specifically, we compared a three-index formulation to a two-index formulation within a general purpose solver. When running these two formulations on a set of benchmark problems, we find that the two-index formulation consistently outperforms the three-index formulation. This highlights the importance of developing proper formulations before developing new heuristic solutions.

While the three-index formulation may have the advantage of clarity in terms of explicitly using one index for each part of the model – the origin, the destination, and the assignment of the company vehicle – the two-index formulation consistently solves more, larger instances to optimality. Of particular note is that we apply both formulations to the two key operating strategies available to providers of designated driver services – fixed pairings between drivers and the company vehicle and flexible movement of drivers among the company vehicles. The two-index formulation is able to solve all instances with flexible routing to optimality, whereas it is only able to solve 128 out of 180 instances to optimality with fixed routing; the three-index formulation performs poorly overall but is surprisingly more successful with fixed routing than flexible.

There is substantial value in having a formulation that solves quickly and reliably for the flexible routing strategy. Our results show that flexibility can bring routing

cost savings of 60% when the capacity allowed in the company vehicles is greater than one. This points to a natural extension of this work with regard to dynamic settings in which customer requests continually arrive over time. In such settings, one would typically employ a rolling horizon approach that runs an optimization model each time new information becomes available. Here, models to solve the problem quickly become more critical.

Another extension to this work revolves around the trade-off between the number of drivers used and the cost of the routes. For example, we see that in the largest case with the most capacity and a flexible routing strategy, an average of 14.3 drivers are needed to serve the jobs with an average cost per job of 65.8. This is in contrast to only nine drivers for a flexible routing strategy with a capacity of one and an average routing cost per job of 131.8. As our objective pertains to the vehicle routes, we did not explicitly take the number of drivers into account. In other settings, it may be necessary to explicitly consider the driver working hours or served customer requests. If, for example, the service provider operates such that the drivers garner tips or remuneration per request served, then paying careful attention to how the drivers are assigned to requests is critical. This gives rise to interesting trade-offs between vehicle-related travel costs and driver-related costs.

References

Agatz N, Bouman P, Schmidt M (2018) Optimization approaches for the traveling salesman problem with drone. *Transportation Science* 52(4):965–981.

Aziez I, Côté JF, Coelho LC (2020) Exact algorithms for the multi-pickup and delivery problem with time windows. *European Journal of Operational Research* 284(3):906–919.

Berbeglia G, Cordeau JF, Gribkovskaia I, Laporte G (2007) Static pickup and delivery problems: a classification scheme and survey. *Top* 15(1):1–31.

Boysen N, Briskorn D, Fedtke S, Schwerdfeger S (2018) Drone delivery from trucks: Drone scheduling for given truck routes. *Networks* 72(4):506–527.

Chung SH, Sah B, Lee J (2020) Optimization for drone and drone-truck combined operations: A review of the state of the art and future directions. *Computers & Operations Research* 123:105004.

Cordeau JF, Laporte G, Ropke S (2008) *Recent Models and Algorithms for One-to-One Pickup and Delivery Problems*, 327–357 (Boston, MA: Springer US), ISBN 978-0-387-77778-8, URL `http://dx.doi.org/10.1007/978-0-387-77778-8_15`.

Cortés CE, Matamala M, Contardo C (2010) The pickup and delivery problem with transfers: Formulation and a branch-and-cut solution method. *European Journal of Operational Research* 200(3):711–724.

Danloup N, Allaoui H, Goncalves G (2018) A comparison of two meta-heuristics for the pickup and delivery problem with transshipment. *Computers & Operations Research* 100:155–171.

Drexl M (2012) Synchronization in vehicle routing-a survey of vrps with multiple synchronization constraints. *Transportation Science* 46(3):297–316.

Editors T (2014) Dudes, tech aim to put an end to duis. URL `https://time.com/27393/dudes-tech-aim-to-put-an-end-to-duis/`.

Fowler GA (2015) Tap your phone, and a designated driver takes you (and your car) home. URL `https://www.wsj.com/articles/tap-your-phone-and-a-designated-driver-takes-you-and-your-car-home-1435174499`.

Furtado M, Munari P, Morabito R (2017) Pickup and delivery problem with time windows: A new compact two-index formulation. *Operations Research Letters* 45, URL `http://dx.doi.org/10.1016/j.orl.2017.04.013`.

Gouveia L, Ruthmair M (2015) Load-dependent and precedence-based models for pickup and delivery problems. *Computers & Operations Research* 63:56–71, ISSN 0305-0548, URL `http://dx.doi.org/https://doi.org/10.1016/j.cor.2015.04.008`.

Gschwind T, Irnich S, Mainz D (2012) Effective handling of dynamic time windows and synchronization with precedences for exact vehicle routing. *Technical report* .

Gurobi Optimization L (2022) Gurobi optimizer reference manual. URL `http://www.gurobi.com`.

Horwitz J (2015) Didi kuaidi, uber's china rival, now offers designated drivers for drunk car owners. URL `https://qz.com/466494/chinas-uber-competitor-is-bringing-on-demand-designated-drivers-to-inebriated-chinese/`.

Karak A, Abdelghany K (2019) The hybrid vehicle-drone routing problem for pick-up and delivery services. *Transportation Research Part C: Emerging Technologies* 102:427–449, ISSN 0968-090X, URL `http://dx.doi.org/https://doi.org/10.1016/j.trc.2019.03.021`.

Macrina G, Pugliese LDP, Guerriero F, Laporte G (2020) Drone-aided routing: A literature review. *Transportation Research Part C: Emerging Technologies* 120:102762.

Maknoon Y, Laporte G (2017) Vehicle routing with cross-dock selection. *Computers & Operations Research* 77:254–266.

Masson R, Lehuédé F, Péton O (2012) Simple temporal problems in route scheduling for the dial–a–ride problem with transfers. *Integration of AI and OR Techniques in Contraint Programming for Combinatorial Optimzation Problems*, 275–291 (Springer).

Masson R, Lehuédé F, Péton O (2014) The dial-a-ride problem with transfers. *Computers & Operations Research* 41:12–23.

Nagy G, Wassan NA, Speranza MG, Archetti C (2013) The vehicle routing problem with divisible deliveries and pickups. *Transportation Science* 49(2):271–294.

NHTSA (2020) National highway traffic safety administration's traffic safety facts annual report: Fatal crashes and percentage alcohol-impaired driving by time of day and crash type, 2020. URL `https://cdan.nhtsa.gov/tsftables/tsfar.htm#`.

Nourinejad M, Zhu S, Bahrami S, Roorda MJ (2015) Vehicle relocation and staff rebalancing in one-way carsharing systems. *Transportation Research Part E: Logistics and Transportation Review* 81:98–113.

Otto A, Agatz N, Campbell J, Golden B, Pesch E (2018) Optimization approaches for civil applications of unmanned aerial vehicles (uavs) or aerial drones: A survey. *Networks* 72(4):411–458.

Parragh SN, Doerner KF, Hartl RF (2008) A survey on pickup and delivery problems. *Journal für Betriebswirtschaft* 58(1):21–51.

Rais A, Alvelos F, Carvalho MS (2014) New mixed integer-programming model for the pickup-and-delivery problem with transshipment. *European Journal of Operational Research* 235(3):530–539.

Ropke S, Cordeau JF (2009) Branch and cut and price for the pickup and delivery problem with time windows. *Transportation Science* 43(3):267–286.

Sang-Hun C (2007) Drinkers in korea dial for designated drivers. URL `http://www.nytimes.com/2007/07/10/world/asia/10korea.html?_r=0`.

Srour FJ, Agatz N, Oppen J (2018) Strategies for handling temporal uncertainty in pickup and delivery problems with time windows. *Transportation Science* 52(1):3–19.

Yu S, Puchinger J, Sun S (2022) Van-based robot hybrid pickup and delivery routing problem. *European Journal of Operational Research* 298(3):894–914, ISSN 0377-2217, URL `http://dx.doi.org/https://doi.org/10.1016/j.ejor.2021.06.009`.