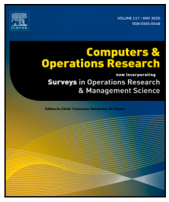
Computers & Operations Research 164 (2024) 106547

Contents lists available at ScienceDirect

Computers and Operations Research

journal homepage: www.elsevier.com/locate/cor



Model formulations for pickup and delivery problems in designated driver services

Alp Arslan [a,*,1], Niels Agatz [b], F. Jordan Srour [c]

[a] Lancaster University Management School, Bailrigg, LA1 4YX, UK
[b] Rotterdam School of Management, Erasmus University, Burgemeester Oudlaan 50, 3062 PA Rotterdam, The Netherlands
[c] Lebanese American University, Department of IT and Operations Management, P.O. Box 13-5053, Chouran, Beirut 1102 2801, Lebanon

ARTICLE INFO

Keywords:
Routing
Dial-a-ride
Pickup and delivery problem
Designated driver services
Routing with precedence

ABSTRACT

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. However, when the drivers may be dropped off by one vehicle and picked up by another, a challenging pick-up and delivery problem arises. In this paper, we study the structural properties of the designated driver problem focusing on the synchronization between company vehicles and drivers. We present a two-index formulations to generate optimal, least-cost routes using a general-purpose solver. We benchmark the two-index formulations against a 3-index formulation and a path enumeration strategy. Based on a set of experiments, we find that the two-index formulation performs well, both in terms of quality and solution time, especially on the formulations with more flexibility in the pairing of drivers to company vehicles. Our computational experiments show that up to 75% 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.

1. Introduction

Drinking and driving is a problem in nearly every country. According to the National Highway Transportation Safety Administration in the United States of America, 13,384 people were killed in alcohol impaired driving crashes in 2021 (Stewart, 2023). 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; for this service the drivers accept donations 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 (Time 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 company cars dedicated to moving single drivers between customer locations. In addition to designated driver services, such companies also run day-time operations targeting business people and people who are unable to drive their own car after medical procedures.

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 and deplete 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 customers' cars at their homes and drive them to the garage for maintenance or repairs.

* Corresponding author.
E-mail addresses: a.arslan1@lancaster.ac.uk (A. Arslan), nagatz@rsm.nl (N. Agatz), jordan.srour@lau.edu.lb (F.J. Srour).
[1] All authors contributed equally to this work.

https://doi.org/10.1016/j.cor.2024.106547
Received 4 August 2023; Received in revised form 11 December 2023; Accepted 10 January 2024
Available online 30 January 2024

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)*.

Companies running designated driver operations, have a choice of how the drivers and company vehicles interact — they may either be coupled in *fixed* pairs or teams or decoupled whereby vehicle routes may *flexibly* pick up or drop off any driver. These two operations lead to three variants of the DDP which we term fixed pairs, fixed teams, and flexible teams. Regardless of the variant, the DDP falls within the realm of Vehicle Routing Problems with Pickup and Delivery, or more specifically Pickup and Delivery Problems with Time Windows (PDPTW). Despite the myriad of PDPTW formulations (Aziez et al., 2020; Furtado et al., 2017; Parragh et al., 2008; Berbeglia et al., 2007), these do not account for the specific complexities of the DDP across all variants.

The contributions of this paper are as follows. First, we formalize the designated driver problem. Second, we propose a two-index formulation that is capable of modeling all DDP variants. Third, we compare the performance of the two-index formulation to a three-index benchmark and a path enumeration strategy while providing insights on the interplay of DDP variant and problem formulation. We also demonstrate the relative merits of the different operating strategies for designated driver services.

The remainder of the paper is organized as follows. The next section provides an overview of the relevant literature. Section 3 gives a formal problem description with notation, highlights the structure of the various operating strategies, provides proofs of NP hardness, and finally presents the two-index formulation for the DDP. Section 4 presents both the three-index and path enumeration formulations used for comparison to the two-index formulation. Section 5 describes our computational experiments and results. The paper concludes with a discussion and suggestions for future work in Section 6.

2. Related literature

Pickup and Delivery problems generally aim to find vehicle routes that serve a set of requests characterized by both pick up and delivery locations while optimizing a set objective or objectives. The Pick-up and Delivery Problem with Time Windows (PDPTW) further includes time windows dictating when service should begin at a request origin and by when the service should be completed at the request destination. The routes must respect the precedence among the paired locations and the time windows at each location. For comprehensive overviews of the literature on pickup and delivery problems, see Parragh et al. (2008) and Berbeglia et al. (2007).

In contrast to the PDPTW, the DDP is a delivery and pickup problem. Following the taxonomy of delivery and pickup problems of Wassan and Nagy (2014), the DDP can be classified as a Vehicle Routing Problem with Mixed Deliveries and Pickups (VRPMDP) in which deliveries and pickups are allowed in any order on a route but at geographically distinct locations. This problem typically arises in the realm of freight transportation and, more specifically, reverse logistics. However, in contrast to the VRPMDP, the goods in the DDP are drivers and are thus not labeled nor tracked as items for delivery or items for pickup. As such, the DDP represents an extension to the VRPMDP whereby the goods picked up at one location may, without restriction, be used to serve the demand at another location.

As the DDP involves the pickup and delivery of people, it also relates to the Dial-a-Ride Problem (DARP), see for recent reviews Mourad et al. (2019), Ho et al. (2018) and Molenbruch et al. (2017). The DARP is a Pickup and Delivery problem with the objective of limiting the inconvenience for the passengers while minimizing operating costs. In contrast to the DARP, the DDP vehicles do not directly transfer customers between their origins and destinations but rather transport the drivers (company employees) between the customer locations. As such, we have a problem setting in which 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 must decide on the routing of the drivers. Depending on the operational strategy of the company, the driver routes may be similar to the company vehicle routes when there is a pairing between drivers and company vehicles. However, when the drivers are not paired to vehicles, the driver routes will be quite different than the vehicle routes.

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. This interplay of routes links the DDP to the stream of literature on vehicle routing problems with route synchronization (Soares et al., 2023; 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. However, continuous and discrete time models through four and three-index formulations were introduced to represent more advanced actions, such as cycles in which vehicles or requests visit a node multiple times, see Pierotti and van Essen (2021).

The synchronization of customer pickup and delivery tasks with driver delivery and pickup movements within the DDP gives rise to various temporal and spatial interdependencies not just in terms of routing, but also in terms of accommodating time windows. Specifically, there are dependencies between the customer time windows for pickup (driver drop-off) and the driver time windows for pick-up (customer drop-off). 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.

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 recognizing that the best suited formulation may vary depending on the operational strategy being used.

We contribute to this line of research by introducing a two-index formulation capable of solving the fundamental driver and vehicle

Table 1
Summary of problem structures.

DDP operations	Description	Two-index reference
Fixed pairs	Drivers are assigned to company vehicles that have a capacity of one.	Yang et al. (2004)
Fixed teams	Drivers are assigned to company vehicles that have a capacity of more than one.	Furtado et al. (2017)
Flexible pairs/teams	Drivers may use any company vehicle; vehicle capacity can be greater than or equal to one.	This work

route structures imposed by operations within the DDP. These three scenarios are summarized in Table 1 along with the citation for the two-index formulations that inspired our overarching formulation. In this work, we show that our formulation for the flexible pairs/teams operation also works, with minor modification, for both the fixed team and fixed pairs settings. We then demonstrate the capability of our two-index formulation to address all three settings in contrast to both a three-index formulation and a path enumeration strategy.

3. The designated driver problem

In this section, we describe the DDP in terms of allowable route structures relative to common operating strategies used in practice. We then demonstrate that in each case, the problem is NP-Hard. We then present the mathematical notation that will serve to model the DDP in the remainder of the paper. Subsequently, we introduce the standard three-index formulation, a path-based formulation, and our two-index formulation.

3.1. Problem structure and complexity

In practice, to simplify planning the routes for company vehicles and drivers, companies often use *fixed pairs* whereby 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 case, we examine a capacity of up to three drivers per vehicle in both a *fixed* and *flexible* mode of operations.

In the *fixed* mode of operations, drivers are assigned to a vehicle upon their departure from the depot. The vehicle then drops off the drivers at the request origins and the same vehicle retrieves the drivers from the request destinations.

In the *flexible* mode of operations, there is no restriction on pairing between drivers and company vehicles. 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. In routing, the drivers can be dropped-off at a request origin by one company vehicle and picked up by either the same or another vehicle.

We now analyze the structure of the different problem variants and their complexity. Focusing on allowable route structures within the general framework of the Pickup and Delivery Problem, we can define three operational settings — immediate, delayed, and decoupled service. Specifically, in the "immediate service" mode, the origin location for a demand must appear on the route immediately prior to the destination location. In contrast, in a "delayed service" routing mode the destination location may be served anywhere in the route after the origin location is served as long as the time windows are obeyed. Finally, in the "decoupled service" version of a pickup and delivery problem, the precedence between the paired demand locations must be respected, but the pick-up and delivery may be served by different vehicles (e.g. on different routes). Each of these three PDPTW variants manifests in the vehicle routing of the DDP depending on the operational strategy adopted. Fig. 1 illustrates, in the left column, potential vehicle routes for the three modes of operation and, in the right column, the driver routes overlaid on these vehicle routes. Depending on the operational strategy of the company, the vehicles will be in one of three routing paradigms — immediate, delayed, or decoupled.

All operational variants of the DDP are NP-hard. Based on above characterizations, we see that the fixed-pair DDP and fixed-team DDP are generalizations of respectively the Pickup and Delivery Problem with Time Windows and multiple vehicles and the Pickup and Delivery Problem with Precedence Constraints and multiple vehicles, both of which are NP-hard problems (Tan and Huang, 2019).

The NP-hard Vehicle Routing Problem with Time Windows (VRPTW) maps to the flexible team DDP as evidenced by the allowed vehicle routes which partition all locations into subsets with Hamiltonian circuits (including a circuit representing job rejections) (Savelsbergh, 1985). The relationship between the VRPTW and DDP is further elaborated through both operational and movement synchronization as described in Soares et al. (2023).

Despite the formal designation of the DDP as NP-Hard, it, like many PDPTW variants, rests on a tractability boundary. For example, with stringent time-windows the DDP may be tractable, but with relaxed time-windows the problem becomes intractable (Tan and Huang, 2019). This makes sense as stringent time-windows serve to dictate an order in which the requests must be served — an order that can be derived through a simple sorting algorithm.

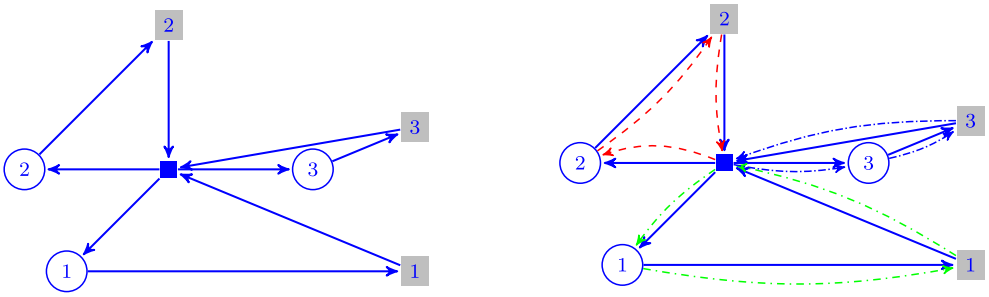
3.2. DDP notation

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.

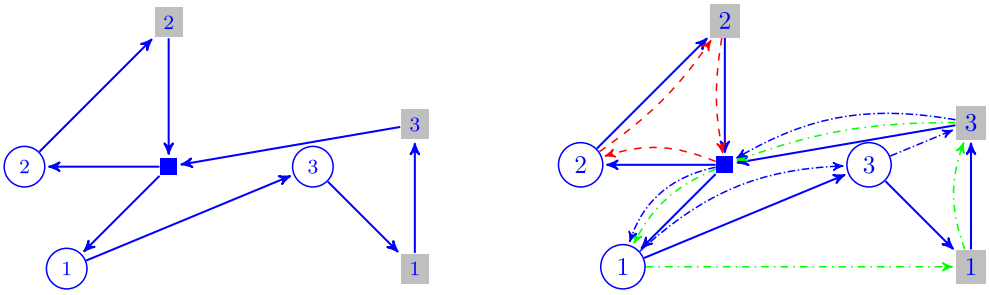
We consider a designated driver service that needs to serve a set $\mathcal{R}$ of customer requests using a team of B drivers and a fleet of homogeneous company vehicles $\mathcal{V}$. Each company vehicle $v \in \mathcal{V}$ starts and ends at the depot and can carry at most Q drivers. Note that company vehicles always have a company driver who remains with the vehicle to drive it 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.

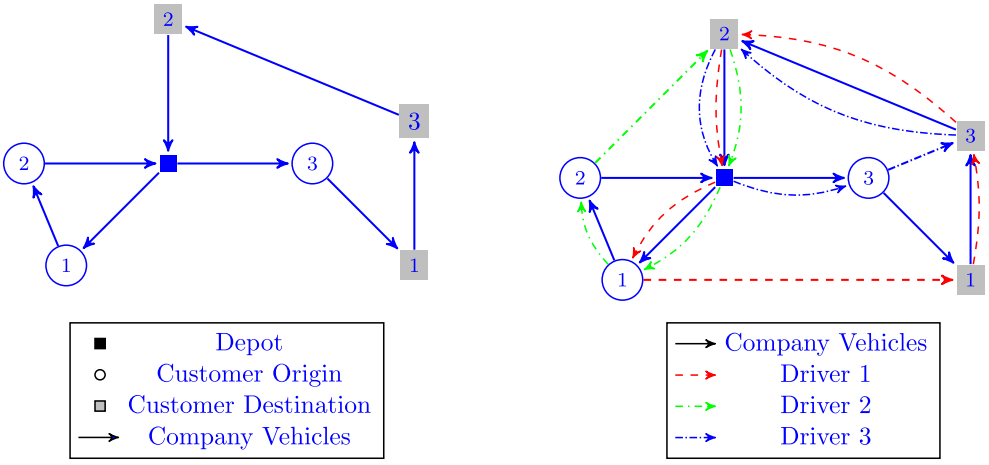
We allow to drop off a driver before the start of the customer time window or pick up the driver beyond the end of service. The time that a driver waits to begin service is only considered the waiting time when the driver is alone; there is no restriction on waiting time when the driver is within the company vehicle. 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



(a): Fixed Pairs Operational Strategy – Drivers assigned to vehicles with capacity of one. Vehicles follow immediate service, PDPTW.



(b): Fixed Teams Operational Strategy – Drivers assigned to vehicles with capacity greater than one. Vehicles follow delayed service, PDPTW.



(c): Flexible Operational Strategy – Drivers not assigned to vehicles with capacity greater than or equal to one; Vehicles follow decoupled service, PDPTW.

Fig. 1. Three routing modes for DDP operations.

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.

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.

3.3. 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. By blending these two formulations, we establish a two-index formulation that can accommodate the fixed pairs, fixed team, and, with the removal of a set of constraints, flexible team DDP variants.

In this formulation, we operate over a graph $G = (\mathcal{N}, \mathcal{E})$, 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 D represents the customer drop-off locations, along with $|\mathcal{V}|$ copies of the depot representing the set of vehicles $\mathcal{V}$ mapped to their location at the point the model is run (in this case, the depot). Thus, the cardinality of $\mathcal{N}$ is $V + 2R$.

In this two-index formulation 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

Table 2
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 $i \in \mathcal{N}$.

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 c_{ij} represents the rejection cost 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 rejected.

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 drivers paired with the company vehicles within the fixed route strategies.

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 leaving node i.

For ease of reference, Table 2 provides a summary of the notation used in the two-subscript formulation.

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

$$\min \sum_{(i,j)\in\mathcal{E}} c_{ij}x_{ij} \tag{1}$$

Constraint sets (2)–(5) 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.

$$\sum_{i\in\mathcal{N}} x_{ij} = 1, \qquad j \in \mathcal{N} \tag{2}$$

$$\sum_{j\in\mathcal{N}} x_{ij} = 1, \qquad i \in \mathcal{N} \tag{3}$$

$$\sum_{i\in\mathcal{N}/o_r} x_{io_r} = y_r \qquad r \in \mathcal{R} \tag{4}$$

$$x_{p_r p_r} = (1 - y_r) \qquad r \in \mathcal{R} \tag{5}$$

Constraint sets (6)–(10) are unique to the fixed team setting as they 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. In the flexible strategy, we remove constraints (6)–(10). Specifically, (6) and (7) 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 (8) and (9) ensure that all jobs on the same route are assigned the same value for v_j. Finally, constraints (10) 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 (7)–(10).

$$v_j \geq j x_{ij} \qquad i \in \mathcal{V};\ \ j \in \mathcal{P} \tag{6}$$

$$v_j \leq j x_{ij} + M(1 - x_{ij}) \qquad i \in \mathcal{V};\ \ j \in \mathcal{P} \tag{7}$$

$$v_j \geq v_i - M(1 - x_{ij}) \qquad i, j \in \mathcal{P} \cup D \tag{8}$$

$$v_j \leq v_i + M(1 - x_{ij}) \qquad i, j \in \mathcal{P} \cup D \tag{9}$$

$$v_{R+i} = v_i \qquad i \in \mathcal{P} \tag{10}$$

Constraints (11)–(12) guarantee that the time that the company vehicle arrives at each location is time feasible. Constraint set (11) ensures that the company vehicle arrives at a node before it departs from that node. Constraint set (12) 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 (12) is the end of the service day.

$$a_j \leq d_j \qquad j \in \mathcal{P} \tag{11}$$

$$a_j \geq d_i + t_{ij} - M(1 - x_{ij}), \qquad i, j \in \mathcal{P} \cup D \tag{12}$$

Constraints (13)–(15) restrict the start time of the service at the origin of a request. Constraint set (13) ensures that service cannot start before a driver has arrived and constraint set (14) makes sure that a driver does not wait alone for longer than allowed before starting service. Constraints (15) make sure that the driver starts service within the service time window. The pick up times of the driver at destination p_r are assigned according to constraints (16) and (17). Constraint set (16) makes sure that the company car picks up the driver before his/her maximum waiting time. Constraint set (17) ensures that the company vehicle does not depart the customer drop-off (driver pickup) location before the driver has arrived.

$$s_r \geq a_{o_r}, \qquad r \in \mathcal{R} \tag{13}$$

$$s_r \leq d_{o_r} + W_{o_r}, \qquad r \in \mathcal{R} \tag{14}$$

$$e_r \leq s_r \leq l_r, \qquad r \in \mathcal{R} \tag{15}$$

$$a_{p_r} \leq s_r + t_r + W_{p_r}, \qquad r \in \mathcal{R} \tag{16}$$

$$s_r + t_r \leq d_{p_r}, \qquad r \in \mathcal{R} \tag{17}$$

We define the number of drivers in each company vehicle at the departure of node i by q_i. Constraints (18) and (19) 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 (20) limits the number of drivers in a company vehicle at any point in time to the maximum capacity Q, while constraint set (21) 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.

$$q_j \leq q_i + A_j + Q(1 - x_{ij}), \qquad i \in \mathcal{N}, j \in \mathcal{P}, i \neq j \tag{18}$$

$$q_j \geq q_i + A_j - Q(1 - x_{ij}), \qquad i \in \mathcal{N}, j \in \mathcal{P}, i \neq j \tag{19}$$

$$\max\{0, A_i\} \leq q_i \leq \min\{Q, Q + A_i\}, \qquad i \in \mathcal{N} \tag{20}$$

$$\sum_{i\in\mathcal{V}} q_i \leq B \tag{21}$$

4. Comparative formulations for the DDP

The two-index formulation is capable of modeling all operational strategies of the DDP. Nevertheless, its success in solving the problem across all strategies varies. To assess the comparative value of the two-index formulation across all operational strategies, we introduce two comparative formulations – a three-index formulation capable of modeling all operational strategies and a path enumeration strategy capable of modeling only the fixed operational strategies.

4.1. Three-index formulation

This formulation follows a traditional three subscript structure (see eg. Ropke and Cordeau, 2009) on a graph $G = (\mathcal{N}, \mathcal{E})$. However, in contrast to the two-index formulation, $\mathcal{N}$ is the union of nodes representing the locations associated with the requests $\mathcal{P} \cup \mathcal{D}$ and a single copy of the depot node 0, where all company vehicles start and end their tours. 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 binary variable y_r, indicates whether request r is served (1) or not (0) and c_r represents the rejection costs.

In addition to the route-related variables, we 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. As in the two-index formulation, we employ variables s_r to capture the time at which drivers start their service from the origin of request r yielding an arrival time at the destination, p_r, of $s_r + t_r$. Finally, to obey the capacity constraint of vehicle v we let q_i^v represent the number of drivers in the company vehicle when leaving node i.

In contrast to the traditional formulation of Ropke and Cordeau (2009), we adapted the objective and constraints in the three-subscript formulation to accommodate job rejections as well as waiting times before the start of service and after the end of service. A full exposition of the three-index model may be seen in Appendix A.

4.2. Path enumeration strategy

Given the precedence between driver drop-off and pick-up tasks, the possible routes in the fixed team setting are somewhat restricted. As such, a standard path enumeration strategy can serve as a viable comparison to our two-index formulation. Note that a standard path enumeration approach is computationally prohibitive for the flexible pair version of the DDP due to the temporal and spatial dependencies between the different routes. This means that it is not possible to decompose the problem along a set of independent paths.

Let Ω be the collection of all feasible paths. A path $\omega \in \Omega$ is defined as the sequence of customer nodes such that $\omega = \{0, n_1, n_2, n_3, \ldots, 0\}$, where $n_i \in \mathcal{N}$ and 0 represents the depot; in the path enumeration strategy the set $\mathcal{N}$ is the union of nodes representing the locations associated with the requests $\mathcal{P} \cup \mathcal{D}$. A path ω is feasible if it drops drivers at the request origins prior to or within the specified time windows (depending on allowable driver waiting time) and picks-up drivers from the corresponding request destinations within the allowed waiting time. The capacity on any path (dictated by the capacity of the vehicle) should also not be violated.

Let $A = [a_{r\omega} | r \in \mathcal{R}, \omega \in \Omega]$ be the resource matrix such that each $a_{r\omega}$ takes a value of one if path ω serves request r. Also, let $\Gamma : \Omega \mapsto \mathbb{Z}^+$ be a function mapping from a path to an integer specifying the minimum number of drivers required to serve the path. Let x_ω be a binary decision variable taking the value of one if path ω is chosen and y_r be a binary variable taking the value of one if request r is served. The cost parameters c_ω and $c_r, r \in \mathcal{R}$ denote the expense of following path ω with a company vehicle and the cost of rejecting request r, respectively.

$$z_{SP} = \min \sum_{\omega \in \Omega} c_\omega x_\omega + \sum_{r \in \mathcal{R}} c_r (1 - y_r) \tag{22}$$

s.t.

$$\sum_{\omega \in \Omega} a_{r\omega} x_\omega = y_r \qquad \forall r \in \mathcal{R} \tag{23}$$

$$\sum_{\omega \in \Omega} x_\omega \leq |\mathcal{V}| \tag{24}$$

$$\sum_{\omega \in \Omega} \Gamma(\omega) x_\omega \leq B \tag{25}$$

The objective (22) minimizes the total cost of the designated driver service. Constraints (23) restrict request r to being served by at most one path while also tracking unserved customers. Constraints (24) and (25) ensure that the solution does not use more vehicles and drivers than the provided fleet.

Observation 1. The path-based formulation given by Eqs. (22)–(25) yields an optimal solution if Ω consists of all non-dominated paths.

Definition 2. Let ω and ω^* be two paths serving the same subset S of customer requests R; $S \subset R$. The path ω^* dominates path ω if $c_{\omega^*} \leq c_\omega \wedge \Gamma(\omega^*) < \Gamma(\omega)$ or $c_{\omega^*} < c_\omega \wedge \Gamma(\omega^*) \leq \Gamma(\omega)$; *i.e.,* path ω^* can serve the same requests as ω with less cost or fewer drivers than path ω.

Details on the construction of paths for inclusion in Ω may be found in Appendix B.

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 four sets of 20 instances for a total of 80 instances based on operational data from a designated driver service company. Each set of 20 instances has a different number of requests: 10, 20, 30, and 40 requests. Within these requests across these instance sets, three, six, nine, and twelve requests, respectively, go from the center of a 100 by 100 square region to the outer areas of the region; two, four, six, and eight requests, respectively, go from the outer areas to the center of the region; and five, ten, fifteen, and twenty requests, respectively, go between randomly selected origins and destinations in the region. In all instances, 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).

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, LLC, 2022) as our IP solver setting a maximum run time of 1800 s (30 min) 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 was as specified in the instances with time windows of 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 drivers available. We run all of the formulations through the instance sets with varying vehicle capacities of one, two, or three, *i.e.*, $Q = 1, 2, 3$.

Table 3
Number of instances solved to optimality within 1800 s, the solve times (median(max)), and percentage optimality gaps (median(max)); fixed operations with five minute time windows and five minutes of wait time allowed.

Cust	Q	# instances solved to optimality			Solve times (s)			Optimality gaps (%)		
		3-Index	2-Index	Path enum.	3-Index	2-Index	Path enum.	3-Index	2-Index	Path enum.
10	1	20	20	20	4.2(9.7)	0.4(0.5)	<0.1	0	0	0
10	2	20	20	20	27.1(95.3)	0.3(0.8)	<0.1	0	0	0
10	3	20	20	20	32.3(74.1)	0.4(0.9)	<0.1	0	0	0
20	1	20	20	20	144.7(758.4)	0.7(3.7)	0.8(1.4)	0	0	0
20	2	0	14	20	n/a	80.8(696.5)	1.4(3.2)	30.1(42.4)	5.7(31.5)	0
20	3	0	13	20	n/a	121.8(613.7)	2.0(5.3)	35.7(50.2)	6.1(28.3)	0
30	1	6	19	20	852.3(1575.1)	76.2(1298.9)	3.3(5.7)	5.1(17.2)	2.9(2.9)	0
30	2	0	1	20	n/a	1295.1(1295.1)	9.2(17.9)	53.7(69.9)	26(45.6)	0
30	3	0	1	20	n/a	974.3(974.3)	13.1(31.3)	65.1(74)	28.1(47.5)	0
40	1	0	5	20	n/a	97.8(316.3)	20.4(38.9)	7.6(14.4)	1.4(3.8)	0
40	2	0	0	20	n/a	n/a	82.1(218.4)	84.5(87.1)	49.3(69.9)	0
40	3	0	0	20	n/a	n/a	117.1(254.1)	84.7(86.4)	50.9(71.9)	0

Table 4
Number of instances solved to optimality within 1800 s, the solve times (median(max)), and percentage optimality gaps (median(max)); flexible operations with five minute time windows and five minutes of wait time allowed.

Cust	Q	# instances solved to optimality		Solve times		Optimality gaps	
		3-Index	2-Index	3-Index	2-Index	3-Index	2-Index
10	1	20	20	23.4 (69.4)	0.1(0.1)	0	0
10	2	20	20	27.1(95.3)	0.1(0.2)	0	0
10	3	20	20	32.3(74.1)	0.1(0.2)	0	0
20	1	4	20	845(1480.5)	0.1(0.2)	6.4 (10.4)	0
20	2	0	20	n/a	0.6(1.0)	31.2 (37.8)	0
20	3	0	20	n/a	0.5(1.4)	29.9 (34.8)	0
30	1	0	20	n/a	0.2(0.2)	5.4 (45.9)	0
30	2	0	20	n/a	5.9(16.6)	45.1 (53.1)	0
30	3	0	20	n/a	3.8(10.7)	43.7 (53.2)	0
40	1	0	20	n/a	3.8(4.7)	11.0(18.9)	0
40	2	0	17	n/a	24.7(90.4)	54.1(59.0)	19.1(24.4)
40	3	0	12	n/a	40.2(1309.2)	54.3(59.2)	3.6(28.1)

5.2. Formulations' computational performances

We begin our study of computational performance by examining the success of all three formulations on the fixed operational strategy.

Table 3 reports the number of instances solved to optimality within 1800 s (30 min), the solution time required if it is less than 1800 seconds, and the median and max optimality gaps of the instances that could not be solved to optimality within 1800 s.

Tables 3 and 4 show that the two-index formulation outperforms the three-index formulation. Using the two-index formulation, we solve significantly more instances to optimality within the maximum run time. In particular, two-index formulation can solve 133 and 229 instances out of all 240 instances for the fixed and flexible team, respectively. On the other hand, the three-index formulation only solves 86 instances within the fixed team setting and 66 instances within the flexible team setting. However, Table 3 also reveals that neither the two- nor three-index formulations are effective for the fixed operational strategies for larger instances. In contrast, the path-based enumeration strategy that selects the least costly set of tours from all non-dominated vehicle tours reaches optimality in all instances and scales well to at least 40 requests.

We now turn our attention to flexible operations where the only comparison possible is that between the two-index to three-index formulations. Table 4 presents the number of instances solved to optimality, the solve times, and the percent gap for the two-index versus three-index formulations applied to the flexible operational setting of the DDP. We observe that the two-index formulation excels on the flexible team cases. Nevertheless, not all instances with 40 requests are solved in 1800 s.

To further explore the performance of the two-index formulation, we also examine the impact of relaxing the time window constraints and allowable driver waiting time constraints. Table 5 presents the computational results for the flexible team setting for allowable waiting times (W_{o_r}, W_{d_r}) and service time windows ($l_r - e_r$) of 5, 10 and 15 min for the instances with 30 and 40 customer requests and the company vehicle's driver capacity (Q) is three. We observe that we can solve all instances with 30 requests, even for the cases with 10 and 15 min timing constraints. However, we see that the two-index formulation cannot find the optimal solutions for all the instances with 40 customer requests. The number of instances solved decreases with the length of the timing constraints. Nevertheless, the two-index formulation still produces near-optimal solutions in most cases. For example, the median optimality gap is between 1.7% and 3.6%. However, for a few instances, the optimality gap remains high, around 30%. These cases coincide with the solutions in which not all customer requests were served, which impacts the lower bound.

5.3. Operational performance

This section presents the operational benefits of flexible teams compared to fixed teams with varying vehicle capacities across the three instance sets. We use our two-index formulation to derive the savings for the flexible team setting and path-based formulation for the fixed team setting. Fig. 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.

The results in Fig. 2 show savings of up to 75% 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

Table 5
Number of instances solved to optimality within 1800 s, the solve times (median(max)), and percentage optimality gaps (median(max)); flexible operations with different time windows and allowable wait times.

Cust	Q	$W_{o_r} = W_{d_r}$	$l_r - e_r$	# Solved to Opt	Comp. time (s)		Optimality gap (%)	
					Median	Max	Median	Max
30	3	5	5	20	3.8	10.7	0	0
30	3	10	10	20	5.5	43.1	0	0
30	3	15	15	20	32.0	386.5	0	0
40	3	5	5	12	40.2	1309.2	3.6	28.1
40	3	10	10	8	91	549.3	1.7	31.7
40	3	15	15	7	236.2	997.2	2.1	33.8

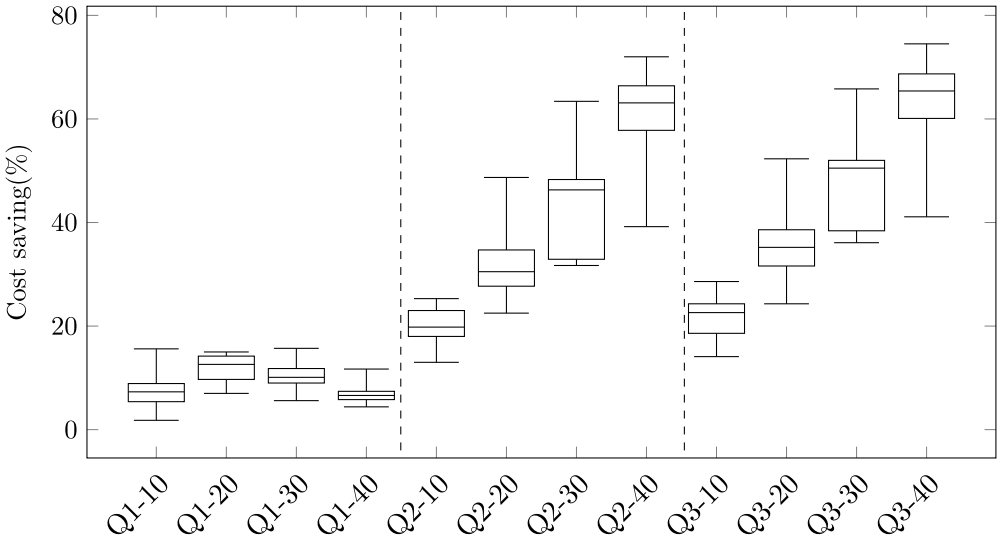


Fig. 2. Cost saving of flexible team as compared to fixed team, $W_{o_r} = W_{d_r} = 5$, Number of instances = 20.

Table 6
Average cost per served order and rejected requests, $W_{o_r} = W_{d_r} = 5$, Number of instances = 20.

Cust	Q	Cost per served request		Rejected	
		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
40	1	269.9	251.9	6.0	5.95
40	2	229.5	78.1	4.8	0.55
40	3	228.3	73.3	4.8	0.55

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 6 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 as long as the fleet and driver capacity allows, 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 7
Average number of deployed company vehicles and drivers, $W_{o_r} = W_{d_r} = 5$, Number of instances = 20.

	Cust	Fixed			Flexible		
		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
	40	9.0	9.0	9.0	9.0	9.0	8.95
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
	40	9.0	12.5	12.6	9.0	14.4	14.9

Table 7 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, 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.

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*. Fig. 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

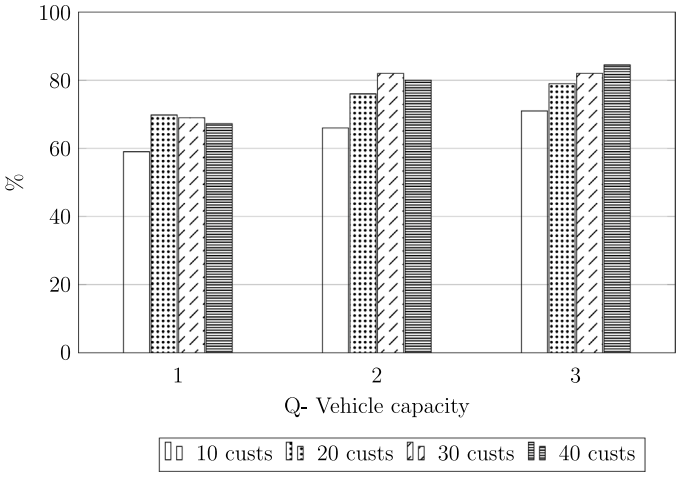


Fig. 3. Percentage of requests served by two different vehicles.

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 $Q = 2$.

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 result is not surprising given the steep difference in the number of decision variables and constraints between the two formulations. This highlights the importance of developing and testing 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 only able to solve 133 out of 240 instances to optimality with fixed routing; the three-index formulation performs poorly overall but is surprisingly more successful with fixed routing than flexible when compared to itself. This result motivates the introduction of a third purpose-built formulation — path enumeration. Using path enumeration we are able to solve all instances considered to optimality. However, when flexible operations strategies are preferred, standard path enumeration is no longer feasible due to the interrelationships between routes. We consider exploring dedicated path enumeration strategies for the flexible team case as a challenging direction for future research.

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 75% 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 14.95 drivers are needed to serve the jobs with an average cost per job of 73.3. 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 251.9. 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.

Table 8
Summary of decision variable 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 $i \in \mathcal{P} \cup \mathcal{D}$.
d_i^v	Departure time of vehicle v from node $i \in \mathcal{P} \cup \mathcal{D}$.
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 $i \in \mathcal{N}$.

CRediT authorship contribution statement

Alp Arslan: Conceptualization, Data curation, Formal analysis, Methodology, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. **Niels Agatz:** Conceptualization, Formal analysis, Investigation, Methodology, Visualization, Writing – original draft, Writing – review & editing. **F. Jordan Srour:** Conceptualization, Formal analysis, Methodology, Visualization, Writing – original draft, Writing – review & editing.

Data availability

Data will be made available on request.

Appendix A. Three-index formulation

For ease of reference, Table 8 presents the notation used in the three-subscript formulation of the DDP.

A.1. The model

The objective (26) minimizes the sum of the cost of serving or rejecting the requests.

$$\min \sum_{v \in \mathcal{V}} \sum_{(i,j) \in \mathcal{E}} c_{ij} x_{ij}^v + \sum_{r \in \mathcal{R}} c_r (1 - y_r) \tag{26}$$

Constraint sets (27)–(29) ensure that each node is visited at most once and that the request origin needs to be visited if the request is served. Constraints (30) make sure that the same vehicle visits both the origin and the destination of a request. We do not need this constraint in the flexible strategy.

$$\sum_{v \in \mathcal{V}} \sum_{i \in \mathcal{N}} x_{ij}^v \leq 1, \quad j \in \mathcal{N} \tag{27}$$

$$\sum_{v \in \mathcal{V}} \sum_{j \in \mathcal{N}} x_{ij}^v \leq 1, \quad i \in \mathcal{N} \tag{28}$$

$$\sum_{i\in\mathcal{N}}\sum_{v\in\mathcal{V}} x^v_{io_r} = y_r \quad r\in\mathcal{R} \tag{29}$$

$$\sum_{i\in\mathcal{N}} x^v_{io_r} - \sum_{j\in\mathcal{N}} x^v_{jp_r} = 0 \quad v\in\mathcal{V},\ r\in\mathcal{R} \tag{30}$$

Constraints (31)–(34) guarantee that the time that the company vehicle arrives at each location is time feasible. Constraint set (31) ensures that the company vehicle arrives to a node before it departs from that node. Constraint set (32) ensures that the arrival to a request origin occurs before the arrival to a destination in order to ensure that a customer pick-up on a route occurs before that customer's drop-off. Constraint sets (33) and (34) 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.

$$a^v_j \le d^v_j \quad j\in\mathcal{P}\cup\mathcal{D},\ v\in\mathcal{V} \tag{31}$$

$$a^v_{p_r} \ge a^v_{o_r}, \quad r\in\mathcal{R}\ v\in\mathcal{V} \tag{32}$$

$$a^v_j \ge d^v_i + t_{ij} - M(1-x^v_{ij}), \quad i,j\in\mathcal{P}\cup\mathcal{D}, v\in\mathcal{V} \tag{33}$$

$$a^v_j \le d^v_i + t_{ij} + M(1-x^v_{ij}), \quad i,j\in\mathcal{P}\cup\mathcal{D}, v\in\mathcal{V} \tag{34}$$

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 (35)–(37) restrict the start time of the service at the origin of a request. Constraint set (35) ensures that service cannot start before a driver has arrived and constraint set (36) makes sure that a driver does not wait alone for longer than allowed before starting service. Constraints (37) 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 (38) and (39). Constraint set (38) makes sure that the company car picks up the driver before his/her maximum waiting time. Constraint set (39) 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.

$$s_r \ge a^v_{o_r} - M(1-x^v_{io_r}), \quad r\in\mathcal{R}, \quad i\in\mathcal{N} \quad v\in\mathcal{V} \tag{35}$$

$$s_r \le d^v_{o_r} + W_{o_r} + M(1-x^v_{io_r}), \quad r\in\mathcal{R}, \quad i\in\mathcal{N} \quad v\in\mathcal{V} \tag{36}$$

$$e_r \le s_r \le l_r, \quad r\in\mathcal{R} \tag{37}$$

$$a^v_{p_r} \le s_r + t_r + W_{p_r} + M(1-x^v_{p_rj}), \quad r\in\mathcal{R}, \quad j\in\mathcal{N}, \quad v\in\mathcal{V} \tag{38}$$

$$s_r + t_r \le d^v_{p_r} + M(1-x^v_{p_rj}), \quad r\in\mathcal{R}, \quad j\in\mathcal{N}, \quad v\in\mathcal{V} \tag{39}$$

We define the number of drivers in company vehicle v at the departure of node i by q^v_i. Constraints (40) and (41) 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 $\mathcal{P}\cup\mathcal{D}$.

Constraint set (42) limits the number of drivers in a company vehicle at any point in time to the maximum capacity Q, while constraint set (43) 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.

$$q^v_j \le q^v_i + A_j + Q(1-x^v_{ij}), \quad i\in\mathcal{N}, j\in\mathcal{P}\cup\mathcal{D}, v\in\mathcal{V}, i\ne j \tag{40}$$

$$q^v_j \ge q^v_i + A_j - Q(1-x^v_{ij}), \quad i\in\mathcal{N}, j\in\mathcal{P}\cup\mathcal{D}, v\in\mathcal{V}, i\ne j \tag{41}$$

$$q^v_i \le Q, \quad i\in\mathcal{N} \tag{42}$$

$$\sum_{v\in\mathcal{V}} q^v_0 \le B \tag{43}$$

A.2. Comparison of two- and three-index formulations

To illustrate the difference between the two-index and three-index formulations, Table 9 presents the number of decision variables and the total number of constraints as a function of the number of requests, R, and the number of vehicles, V for the fixed and the flexible team setting.

Unsurprisingly, the two-index formulation requires fewer decision variables by dropping the vehicle index. This means that the number of decision variables for the three-index formulation is approximately a factor V larger. For the same reason, the two-index formulation has fewer constraints than the three-index formulation.

Comparing the fixed team and the flexible team setting, we see that we need the same (three-index) or fewer (two-index) decision variables in the flexible team setting. The reduction of variables in the two-index formulation is possible as we no longer need to track the sequence of each vehicle's visit via v_j. In both formulations, we can reduce the number of constraints as the flexible team setting is less restrictive.

Appendix B. Enumeration of paths

The path-based enumeration strategy requires determining all feasible customer subsets with non-dominated paths to reach the optimal solution. In this section, we describe a recursive algorithm to enumerate the collection of paths that form the set Ω. The recursive algorithm starts from subsets with only one request and then increases the subset sizes gradually. Inspired by the study of Stiglic et al. (2015), we use the following observations to enumerate the subsets.

Let $\mathcal{R}_i$ be the subset of $\mathcal{R}$ with cardinality $i\in\mathbb{Z}^+$.

Observation 2. A subset of $\mathcal{R}$ has a feasible path if the breadth-first labeling algorithm produces at least one path.

Observation 3. If subset $\mathcal{R}_s$ has at least one feasible path, then each subset $\mathcal{R}_m\subset\mathcal{R}_s$ has at least a feasible path.

Observation 3 implies that we do not need to enumerate a customer subset with s customers if one of its subsets does not have any feasible path.

B.1. Breadth first labeling algorithm to find non-dominated paths for a customer set

This section describes the algorithm to find non-dominated vehicle paths to serve a given set of customers or certify that a single vehicle cannot serve the customers in a fixed team setting.

We use a label-setting algorithm built on the partial path definition. The algorithm extends each partial path by one node checking if the addition forms a feasible complete path or becomes infeasible. We define a partial path τ and its label $L(\tau) = \{N^{Seq}, t^a, t^d, T^{ser}, N^{Imm}, N^{Nec}, q\}$, where

1. $N^{Seq} = \{n_{[1]},\dots,n_{[l]}\}$: sequence of nodes, in order, from the first $n_{[1]}$ to the last visited node $n_{[l]}$.

Table 9
Number of decision variables and constraints, R: **number of requests**, V: **number of vehicles**.

Teams	Metric	Three-index formulation	Two-index formulation
Fixed	No. of Decision Variables	$4R^2V + 10RV + 2V + 2R$	$4R^2 + 4RV + V^2 + 10R + V$
	No. of Constraints	$20R^2V + 9RV + 9R + V + 4$	$16R^2 + 4RV + 15R + 5V$
Flexible	No. of Decision Variables	$4R^2V + 10RV + 2V + 2R$	$4R^2 + 4RV + V^2 + 9R + V$
	No. of Constraints	$20R^2V + 8RV + 9R + V + 4$	$14R^2 + 2RV + 14R + 5V$

2. t^a, t^d: time of arrival and departure, respectively, associated with the vehicle arriving and departing the last node, $n_{[l]}$.
3. $T^{Ser} = \{(a_{r_{[1]}}, s_{r_{[1]}}), \dots\}$: vehicle arrival and service start time for a request served in order $[i]$.
4. $N^{Imm} = \{n_i, n_j, \dots\}$: the set of nodes that can be reached immediately from node $n_{[l]}$
5. $N^{Nec} = \{n_i, n_j, n_m, n_s \cdots\}$: all nodes that are necessary to form a feasible path.
6. q: maximum number of drivers required to serve all nodes until node $n_{[l]}$

We use a similar logic of label extension described in the study by Feillet et al. (2004). That is, we start with a null partial path, such as $L(\emptyset) = \{V^{Seq} = \emptyset, 0, 0, T^{Ser} = \emptyset, V^{Imm} = \{o_{r_1}, o_{r_2}\}, V^{Nec} = \{o_{r_1}, o_{r_2}, d_{r_1}, d_{r_2}\}, 0\}$ for the customer set of $\{r_1, r_2\}$. We explore all extension possibilities over the set V^{Imm} for all feasible partial path possibilities. If the extension is feasible, we form a new partial path, including the node where the path is extended and keep it for further extensions. If the extension is not feasible, the extended partial path will be pruned from the search.

For the label extension for the partial path τ with label $L(\tau)$ to extend the node n, we carry out the following equations:

1. $N^{Seq*} = N^{Seq} \cup \{n\}$, where new node is added to sequence,
2. $t^{a*} = t^d + t_{n_{[l]}v}$, where the arrival time to node n,
3. $t^{d*} = \begin{cases} \max\{t^{a*}, e_n - W_{o_r}\} \text{ if } n \in \mathcal{P} \\ \max\{t^{a*}, s_{\rho(n)} + t_{o_{\rho(n)}n}\} \text{ if } n \in \mathcal{D} \end{cases}$, where $\rho(n) : \mathcal{N} \mapsto \mathcal{R}$ is a function that returns the request index of node n,
4. $T^{Ser*} = T^{Ser} \cup \{(t^{a*}, \max\{t^{a*}, e_n - W_{o_r}\})\}$, if $n \in \mathcal{P}$, where the service start time for the customer origin node is recorded and stacked in order of visit
5. $N^{Imm*} = \begin{cases} N^{Imm} - \{n\} \cup \{d_{\rho(n)}\} \text{ if } n \in \mathcal{P} \\ N^{Imm} - \{n\} \text{ if } n \in \mathcal{D} \end{cases}$,
6. $N^{Nec*} = N^{Nec} - \{n\}$, the visited node is discarded,
7. $q*$, if partial path driver requirement increased by one.

Two feasibility checks are required for each label extension: (i) time feasibility and (ii) driver number feasibility. The latter checks whether the partial path reaches a new customer origin consecutively more than Q times or not. The former check is unique to the DDP due to the parallel time tracking of the company vehicle and drivers' service starting and ending times. We need to go through several cases to determine the feasibility depending on whether the node n is a pick-up or drop-off node as follows.

$n \in \mathcal{P}$, **Pick-up node**. If the company vehicle arrival time to node n, *i.e.,* t^a, is greater than l_r; then the partial path is infeasible. This condition means that the vehicle arrives at the customer's origin later than the request's time window.

$n \in \mathcal{D}$, **Drop-off node**. If the company vehicle arrival time to n, *i.e.,* t^a, is in the following range:

- $t^a > s_{\rho(n)} + t_{o_{\rho(n)}(n)n} + W_{d_{\rho(n)}} + W_{o_{\rho(n)}}$ then the partial path is infeasible. This condition means that the company vehicle arrives at the customer destination later than the allowed time limits.
- $s_{\rho(n)} + t_{o_{\rho(n)}n} + W_{d_{\rho(n)}} + W_{o_{\rho(n)}} \geq t^a \geq s_{\rho(n)} + t_{o_{\rho(n)}n} + W_{d_{\rho(n)}}$ then, a special routine needs to be called to check for infeasibility. Verbally, this condition specifies that the driver waits at the destination more than the allowed $W_{d_{\rho(n)}}$. However, it may be possible to adjust the driver's arrival time to node n by delaying the service start time at the customer origin, and the partial path could be feasible. In this routine, we adapted the seminal idea from the study by Savelsbergh (1985). That is, we search for the minimum shift of company vehicle departure from the origin node of request $\rho(n)$. This means that after at most the number of visited nodes between node n and the origin of request $\rho(n)$; *i.e.,* $[n] - [o_{\rho(n)}]$ steps, we can conclude whether the partial path is infeasible or not.

The breadth-first labeling algorithm finalizes whether there is no partial path to extend before reaching all nodes, or if the algorithm generates at least a single complete path. If multiple complete paths are generated, we can eliminate some according to the dominance rule provided by Definition 2. The remaining non-dominated paths are added to set Ω to solve the path-based formulation given in Eqs. (22)–(25).

References

Aziez, I., Côté, J.-F., Coelho, L.C., 2020. Exact algorithms for the multi-pickup and delivery problem with time windows. European J. Oper. Res. 284 (3), 906–919.

Berbeglia, G., Cordeau, J.-F., Gribkovskaia, I., Laporte, G., 2007. Static pickup and delivery problems: A classification scheme and survey. Top 15 (1), 1–31.

Cortés, C.E., Matamala, M., Contardo, C., 2010. The pickup and delivery problem with transfers: Formulation and a branch-and-cut solution method. European J. Oper. Res. 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. Comput. Oper. Res. 100, 155–171.

Drexl, M., 2012. Synchronization in vehicle routing-a survey of VRPs with multiple synchronization constraints. Transp. Sci. 46 (3), 297–316.

Feillet, D., Dejax, P., Gendreau, M., Gueguen, C., 2004. An exact algorithm for the elementary shortest path problem with resource constraints: Application to some vehicle routing problems. Netw.: Int. J. 44 (3), 216–229.

Fowler, G.A., 2015. Tap your phone, and a designated driver takes you (and your car) home. Wall Street J. 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. Oper. Res. Lett. 45, 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. Comput. Oper. Res. 63, 56–71. http://dx.doi.org/10.1016/j.cor.2015.04.008, URL https://www.sciencedirect.com/science/article/pii/S030505481500088X.

Gschwind, T., Irnich, S., Mainz, D., 2012. Effective Handling of Dynamic Time Windows and Synchronization with Precedences for Exact Vehicle Routing. TeChnical Report, Johannes Gutenberg University Mainz, Germany.

Gurobi Optimization, LLC, 2022. Gurobi optimizer reference manual. URL http://www.gurobi.com.

Ho, S.C., Szeto, W., Kuo, Y.-H., Leung, J.M., Petering, M., Tou, T.W., 2018. A survey of dial-a-ride problems: Literature review and recent developments. Transp. Res. B.

Horwitz, J., 2015. Didi Kuaidi, Uber's China Rival, Now Offers Designated Drivers for Drunk Car Owners. Quartz, URL https://qz.com/466494/chinas-uber-competitor-is-bringing-on-demand-designated-drivers-to-inebriated-chinese/.

Maknoon, Y., Laporte, G., 2017. Vehicle routing with cross-dock selection. Comput. Oper. Res. 77, 254–266.

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

Masson, R., Lehuédé, F., Péton, O., 2014. The dial-a-ride problem with transfers. Comput. Oper. Res. 41, 12–23.

Molenbruch, Y., Braekers, K., Caris, A., 2017. Typology and literature review for dial-a-ride problems. Ann. Oper. Res. 259, 295–325.

Mourad, A., Puchinger, J., Chu, C., 2019. A survey of models and algorithms for optimizing shared mobility. Transp. Res. B 123, 323–346.
Nourinejad, M., Zhu, S., Bahrami, S., Roorda, M.J., 2015. Vehicle relocation and staff rebalancing in one-way carsharing systems. Transp. Res. Part E: Logist. Transp. Rev. 81, 98–113.
Parragh, S.N., Doerner, K.F., Hartl, R.F., 2008. A survey on pickup and delivery problems. J. für Betriebswirtschaft 58 (1), 21–51.
Pierotti, J., van Essen, J.T., 2021. MILP models for the dial-a-ride problem with transfers. EURO J. Transp. Logist. 10, 100037.
Rais, A., Alvelos, F., Carvalho, M.S., 2014. New mixed integer-programming model for the pickup-and-delivery problem with transshipment. European J. Oper. Res. 235 (3), 530–539.
Ropke, S., Cordeau, J.-F., 2009. Branch and cut and price for the pickup and delivery problem with time windows. Transp. Sci. 43 (3), 267–286.
Sang-Hun, C., 2007. Drinkers in Korea Dial for Designated Drivers. New York Times, URL http://www.nytimes.com/2007/07/10/world/asia/10korea.html?_r=0.
Savelsbergh, M.W., 1985. Local search in routing problems with time windows. Ann. Oper. Res. 4, 285–305.
Soares, R., Marques, A., Amorim, P., Parragh, S.N., 2023. Synchronisation in vehicle routing: Classification schema, modelling framework and literature review. European J. Oper. Res..
Srour, F.J., Agatz, N., Oppen, J., 2018. Strategies for handling temporal uncertainty in pickup and delivery problems with time windows. Transp. Sci. 52 (1), 3–19.
Stewart, T., 2023. Overview of Motor Vehicle Traffic Crashes in 2021 (Report No. DOT HS 813 435). National Highway Traffic Safety Administration, URL https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/813435.
Stiglic, M., Agatz, N., Savelsbergh, M., Gradisar, M., 2015. The benefits of meeting points in ride-sharing systems. Transp. Res. B 82, 36–53.
Tan, X., Huang, J.X., 2019. On computational complexity of pickup-and-delivery problems with precedence constraints or time windows. In: IJCAI. pp. 5635–5643.
Time Editors, 2014. Dudes, Tech Aim to Put an End to DUIs. Time, URL https://time.com/27393/dudes-tech-aim-to-put-an-end-to-duis/.
Wassan, N.A., Nagy, G., 2014. Vehicle routing problem with deliveries and pickups: Modelling issues and meta-heuristics solution approaches. Int. J. Transp. 2 (1), 95–110.
Yang, J., Jaillet, P., Mahmassani, H., 2004. Real-time multivehicle truckload pickup and delivery problems. Transp. Sci. 38 (2), 135–148.