OR Spectrum
https://doi.org/10.1007/s00291-024-00754-2

ORIGINAL ARTICLE

Optimizing combined tours: The truck-and-cargo-bike case

Philine Schiewe[1] · Moritz Stinzendörfer[2]

Received: 25 August 2023 / Accepted: 2 February 2024
© The Author(s) 2024

Abstract
In this paper, we introduce a last-mile delivery concept that is well suited for urban areas. By jointly optimizing the tour of a truck and a cargo bike, we ensure that each vehicle is used optimally. Here, we assume that the bike is restocked by meeting up with the truck so that no dedicated mini-hubs have to be constructed. We model different objective functions and analyze the different variants in comparison to the traveling salesperson problem as well as the capacitated vehicle routing problem. In an experimental evaluation, we compare MIP formulations for different problem variants and assess several heuristic approaches to solve large-scale instances. These results show that we can outperform the truck-only delivery in terms of completion time while reducing the distance driven by the truck.

Keywords Last-mile logistics · Cargo bike · Mixed-integer programming · Two-echelon · Routing

1 Introduction

Last-mile deliveries in urban areas are becoming increasingly important, as highlighted, for example, in Boysen et al. (2020). With a given set of demand points, the goal is to determine routes for delivery vehicles that cover all demand points and minimize a generalized cost function. To avoid straining the urban infrastructure, even more, fast, efficient and environmentally friendly solution approaches are

This work was partially supported by DFG under SCHO 1140/8-2 and by the Bundesministerium für Wirtschaft und Energie (BMWi) Grant Number 01MK20013A.

✉ Moritz Stinzendörfer
stinzendoerfer@mathematik.uni-kl.de; m.stinzendoerfer@math.rptu.de

Philine Schiewe
philine.schiewe@aalto.fi

[1] Systems Analysis Laboratory Department of Mathematics and Systems Analysis, Aalto University, Otakaari 1, 02150 Espoo, Finland

[2] Department of Mathematics, RPTU Kaiserslautern-Landau, Paul-Ehrlich-Str. 14, 67663 Kaiserslautern, Germany

Published online: 10 March 2024

needed. Thus, recent approaches focus on supplementing truck deliveries with additional transport modes with a lower environmental impact that is well suited in an urban setting. Some approaches include drones, other small autonomous vehicles or cargo bikes. Here, each kind of vehicle has its own properties and restrictions, resulting in differing modeling approaches. Drones, e.g., have a very limited capacity, but can be transported by a delivery truck, while cargo bikes have a significantly higher capacity, but usually cannot be transported themselves. By distributing deliveries to multiple vehicles, the need to hand over packages from one (larger) vehicle to the next (smaller) one arises. This can either take place at dedicated hubs, which might also serve as mini-depots, or at any location along the route of the larger vehicle. Introducing hubs as mini-depots allows for scheduling the varying delivery vehicles almost independently, but requires allocating suitable locations and implies building costs. Alternatively, handing over deliveries along the route requires synchronization between the routes of different vehicles, which increases flexibility in response to demand fluctuations, but results in a more involved model.

In this paper, we model and solve the problem when deliveries are combined between a truck and a cargo bike, and goods can be handed over at any stop on the route. To be more precise, we have a set of customers, each with a specific demand, a truck depot used as a starting point for a delivery truck and storing all packages corresponding to the customer's demand, as well as a bike depot that serves as the starting point for one or multiple cargo bikes. Now, each vehicle supplies a subset of customers such that all demands of the complete set of customers are fulfilled. Due to the limited capacity of the cargo bike and the fact that the truck transports all packages, the bike has to meet up with the truck regularly to be restocked. This can be done at any customer location while the truck supplies the corresponding customer. Consequently, the objective is to minimize the resulting generalized costs depending on time or distance. We illustrate the concept of combine truck-and-cargo-bike tours in Example 1.

Example 1 Figure 1 shows a truck tour (solid edges) and a bike tour (dashed edges). Both tours start and end at the respective depots. (Black house in the middle corresponds to the cargo bike depot, and truck depot is on the left.) All other nodes represent customer locations with unit demand. As the cargo bike has a capacity of two, bike and truck tours are synchronized accordingly, such that the first and the fifth customer location of the truck tour serve as handover locations.

Assuming unit edge length, the truck tour has a length of nine, while the bike tour has a length of eight due to the synchronization at the second handover location. The completion time, i.e., the time until both vehicles have returned to the depot, is nine.

As no dedicated infrastructure has to be built for this delivery setting, this method can easily be implemented and is robust against changing demand. The main contributions of the paper can be summarized as follows:

- we formulate the main problem and introduce MIP formulations for several variants with different cost functions, objectives and multiple cargo bikes,

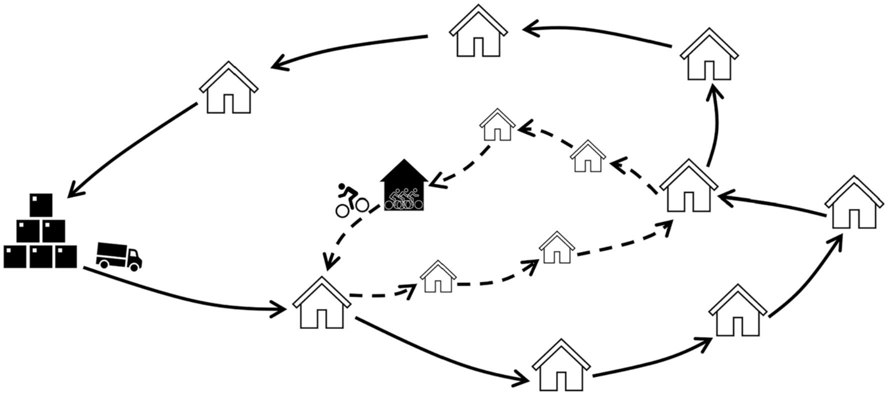


Fig. 1 Example of truck (solid edges) and bike tour (dashed edges) with bike capacity two and two hand-over locations

- we discuss the relationship to the traveling salesperson problem (TSP) as well as the capacitated vehicle routing problem, analyze the potential improvements compared to the TSP and provide approximation bounds,
- we propose three solution algorithms based on clustering, savings and reinforcement learning,
- and we evaluate the performance of all algorithms and MIP formulations on artificial and close-to-real-world instances.

In Sect. 2, we discuss the relevant literature and differentiate our setting from existing papers. We model the problem as a mixed-integer program in Sect. 3 and introduce several possible objective functions as well as problem variants. Three heuristic solution approaches are introduced in Sect. 4 and evaluated experimentally on artificial and close-to-real-world instances in Sect. 5. Here, we analyze how the instance size, the bike capacity and the speed differences between the vehicles influence the solution process and the structure of the solutions. Section 6 concludes the paper.

2 Literature review

City logistics has gained increasing attention in recent years as urbanization continues to intensify. It typically refers to the planning, organization and optimization of freight transportation and logistics activities within urban areas and includes the movement of goods, services and information (Bektas et al. 2015). Addressing the complex interactions between various stakeholders such as suppliers, carriers, retailers and consumers, the integration of advanced technologies, such as intelligent transport systems, big data and artificial intelligence, plays a crucial role in more recent approaches (Taniguchi et al. 2020). In this work, we address some of the arising key challenges, including environmental sustainability, congestion and traffic management, and last-mile delivery (Anand et al. 2012; Russo and Comi 2020), with a focus on the latter.

There are many different types of last-mile delivery concepts in the literature; for an overview, we refer to Boysen et al. (2020). Often, these concepts are variants or extensions of the *traveling salesperson problem* (TSP) (Jünger et al. 1995; Applegate et al. 2011) or *vehicle routing problem* (VRP) (Toth and Vigo 2014). Some of those include time windows (Desrochers et al. 1992), backhauls (Goetschalckx and Jacobs-Blecha 1989), split deliveries (Dror and Trudeau 1990), stochastic demands (Bertsimas 1992), stochastic presence of customers (Gendreau et al. 1995) or stochastic travel times (Laporte et al. 1992; Malandraki and Daskin 1992). In other variants, a mixed fleet is considered, called *heterogeneous fleet VRP* (Baldacci et al. 2008), where the types of vehicles can differ in capacity, speed, variable and fixed costs, and the customers that they can access. For example, the *truck-and-trailer routing problem* (Chao 2002; Lin et al. 2011) involves managing a fleet comprising at least two vehicle types: normal trucks without trailers and truck-and-trailer combinations. While the latter is attractive due to its larger overall capacity, some customers can only be reached by a normal truck. A more general version of this is the *VRP with Trailers and Transshipments* (Drexl 2013) where there is no fixed assignment of trailers to trucks.

Another important variant is the *pickup and delivery problem* (Berbeglia et al. 2007), which can be expanded by integrating a fixed route service as in Ghilas et al. (2016). Here, requests need to be scheduled involving pickup and drop-off locations with specified time windows and maximum ride times. The orders can be fulfilled by a delivery truck alone or be supplemented by a fixed route service, meaning that it can cover a part of the route and make use of spare capacity. The goal is to allocate vehicles, plan routes efficiently, and meet capacity and time window constraints while considering both trucks and fixed route services.

In the *vehicle routing problem with cross-docking* (Wen et al. 2009), a given set of pickup and delivery requests is addressed by a fleet of identical vehicles starting and ending at a cross-dock. They are used to drive both a pickup route to collect goods from suppliers and transport them to the cross-dock and, after unloading and reloading, a delivery route to deliver them to the corresponding customers.

The authors of Zäpfel and Bögl (2008) tackle the problem of local letter mail distribution. This involves simultaneous vehicle and driver routing and scheduling, taking into account constraints on working and driving times and considering the option of outsourcing vehicle routes to external carriers. Here, delivery routes, transporting shipments from the distribution center to the local post offices, as well as separate pickup routes for outbound shipments, have to be planned during one week, allowing the reuse of drivers and vehicles across multiple routes.

Our proposed approach corresponds to the field of vehicle routing problems with multiple synchronization constraints. Besides the assignment of customers to the supplying vehicle, additional synchronization requirements in terms of time, location and load are necessary to model the related problems. Typically, there are different types of autonomous and non-autonomous vehicles, capacities, tasks or loads, and locations, such as customers, transfer locations or depots. For interested readers, we refer to Drexl (2012). The main difference between our approach to those in

Drexl (2012) lies in the joint consideration of large (truck) and small (cargo bikes) independent vehicles, both supplying customers that can serve as transfer locations and their synchronization in time.

Our model is related to both models for routing trucks and drones and two-echelon routing problems. Table 1 gives an overview of similarities and differences.

The *flying sidekick traveling salesman problem* (FSTSP) is introduced in Murray and Chu (2015) and consists of optimal customer assignments for a drone supporting a delivery truck. In each drone subtour, starting at the depot or from a customer location after synchronizing with the truck, the drone supplies exactly one customer within the limited flight endurance. For longer distances, recharging or to conserve battery power, the drone can be transported by the truck. In order to minimize the time required to serve all customers, the authors proposed an MIP formulation as well as a route and re-assign heuristic. Another simple greedy heuristic to solve FSTSP is presented in Crişan and Nechita (2019). Starting from a TSP tour, nodes are assigned to the drone in descending order of the corresponding time savings.

The authors of Agatz et al. (2018) propose the *TSP with drone* (TSP-D), a similar concept to FSTSP. Here, the truck can visit customers more than once as this could be useful for reloading or transporting the drone and can wait for the drone to return to the same node where the drone started. They present an operation-based IP formulation, where an operation represents part of a tour that contains at most one drone node. Since the number of operations grows exponentially with the number of nodes, the IP can only be solved for small instances, and the authors introduce a route-first cluster-second heuristic. In Bouman et al. (2018), the same authors presented an improved method as well as a variant that considers a subset of operations in order to reduce the computational time at the expense of accuracy.

Two alternative MIP formulations to the FSTSP are provided in Schermer et al. (2020). In fact, the authors propose a third MIP formulation, partly based on the concept of operations introduced in Agatz et al. (2018), with an exponential number of constraints and use this formulation for a branch-and-cut approach.

A slightly different concept regarding the combination of trucks and drones is pursued in Amorosi et al. (2021). Instead of visiting nodes in the graph, a given percentage of the edge lengths of a set of graphs has to be inspected (i.e., visited) by a drone. To address this problem, the authors propose a nonlinear MIP formulation and a matheuristic.

In *two-echelon routing problems* (a special case of *multi-echelon vehicle routing problems* (Gonzalez-Feliu et al. 2008; Perboli et al. 2011)), a distribution network with two levels (echelons) is considered; see (Cuda et al. 2015; Sluijk et al. 2023). Here, the primary vehicles start from a subset of predefined depots and transport goods to a subset of predefined handover locations, called satellites, with their own capacities. From there, the secondary vehicles deliver the goods to the customers. While in some formulations, the subsets of depots and satellites have to be selected (e.g., Contardo et al. 2012), in others, one (Nguyen et al. 2012) or both (Hemmelmayr et al. 2012) are predefined. There are also variants where both vehicles have to be at a satellite location at the same time, as there is no way to store the goods (Grangier et al. 2016). A variant of the two-echelon routing problem concerning trucks and cargo bikes, similar to the concept we introduce, can be found in

Table 1 Overview of the related literature

Literature	PV supplies customers	Multiple SVs per PV possible	SV moves on network	SV can be transported	Handover at customer location	Synchronization of PV and SV	SV capacity > 1 possible	Predefined depot/handover/assignment
Murray and Chu (2015)	•	◦[a]	◦	•	•	•	◦	•/◦/◦
Agatz et al. (2018)	•	◦[a]	◦	•	•	•	◦	•/◦/◦
Liu et al. (2020)	•	◦[a]	•	•	•	•	•	•/◦/◦
Amorosi et al. (2021)	◦	•[a]	◦	•	◦[c]	•	◦[g]	•/◦/•
Anderluh et al. (2017)	•	•[b]	•	◦	◦[d]	•	•	•/•/•
Grangier et al. (2016)	◦	•	•	◦	◦[d]	•	•	•/•/•
Nguyen and Hà (2023)	•	•[a]	◦	◦	◦[e]	◦	•[h]	•/-/◦
Boysen et al. (2018)	•	•[a]	•	•	•	•	◦	•/◦/•[i]
Contardo et al. (2012)	◦	•	•	◦	◦[d]	◦	•	•[j]/•/•
Nguyen et al. (2012)	◦	•	•	◦	◦[d]	◦	•	•/•/•
Hemmelmayr et al. (2012)	◦	•	•	◦	◦[d]	◦	•	•/•/•
Anderluh et al. (2021)	•	•	•	◦	◦[d]	•	•	•/•/•[k]
Li et al. (2021)	◦	•	•	◦	◦[d]	•[f]	•	•/•/•
Li et al. (2022)	•	◦	•	•	•	•	•	•/◦/•[l]
Tong et al. (2022)	•	◦[a]	•	•	•	•	◦	•/◦/◦
Bogyrbayeva et al. (2023)	•	◦[a]	•	•	•	•	◦	•/◦/◦
Sutrisno and Yang (2023)	◦	•	•	◦	◦[d]	•	•	•/•[j]/◦
Zhou et al. (2023)	•	•[a]	•	•	•	◦	◦	•/◦/◦
Vu et al. (2022)	◦	•[a]	•	•	◦[d]	◦	◦	•/•[j]/•
our approach	•	•[b]	•	◦	•	•	•	•/◦/◦

Notation: yes (•), no (◦), primary vehicle (PV), secondary vehicle (SV)

[a] Drone

[b] Cargo bike

[c] Anywhere

[d] Satellite

[e] Depot

[f] Synchronization between two types of satellites

[g] One target graph per drone

[h] Only one customer per drone tour

[i] Predefined truck tour

[j] Predefined set of possible depots or handover locations

[k] Predefined assignment, except for a subset of customers in a so-called gray zone

[l] Predefined set of customers that cannot be visited by the primary vehicle, the remaining customers are visited by the secondary or both vehicles

Anderluh et al. (2017). As in our concept, both vehicles start at their corresponding depot, deliver packages to customers and require synchronization in time and location to reload the cargo bike. The main difference to our approach lies in the fact that the truck and bike nodes are predefined in Anderluh et al. (2017), reducing the complexity. Moreover, the possible locations to reload the cargo bike differ from the customer nodes and are determined a priori. In Anderluh et al. (2021), the authors extend the aforementioned model by allowing a so-called gray zone in which the corresponding customers can be supplied by both types of vehicles.

In Appendices B.2 and B.1, we discuss the relationship to truck-and-drone models as well as two-echelon routing models in more detail by showing how our model can be adapted to the problem described in Murray and Chu (2015); Anderluh et al. (2017).

3 Model

To introduce our problem formally, we use the following notation. Let $G' = (V, E')$ be a digraph, where $v_t \in V$ and $v_b \in V$ represent the truck depot and the bike depot, respectively. The remaining nodes $v_1, ..., v_n \in V$ serve as the customer locations with demand $d(v_i) \in \mathbb{R}_{\geq 0}$ (w.l.o.g. $d(v_i) > 0$) for all $i \in [n]$. We set $d(v_t) = d(v_b) = 0$ for completeness. Since we only use one truck, we assume that $\sum_{v \in V} d(v) \leq C_t$, where C_t denotes the truck capacity and C_b the bike capacity. Note that the demand at each node has to be served completely by one vehicle.

For each edge $e = (v_i, v_j) \in E'$, we have truck $c^t(e) = c^t(v_i, v_j)$ and bike weights $c^b(e) = c^b(v_i, v_j)$ corresponding to the costs of the truck and the bike to travel from v_i to v_j, respectively. To simplify the notation for the remainder of the paper, we construct a complete digraph $G = (V, E)$ from G' and define $c^t(x, y) := c^t(sp^t(x, y))$ and $c^b(x, y) := c^b(sp^b(x, y))$. Here $sp^t(x, y)$ denotes the shortest path from x to y (in terms of truck costs c^t in G') and $sp^b(x, y)$ the shortest path in terms of bike costs c^b in G'. If there exists no path between two nodes x' and y' for one of the two vehicles, we define $c^t(x', y') := \infty$ or $c^b(x', y') := \infty$. As a consequence, both cost functions c^t and c^b in the modified graph G satisfy the triangle inequality. To transfer a solution back to the original graph G', we store the corresponding shortest paths.

We call $(\mathcal{T}, \mathcal{B})$ a solution to the combined logistics problem, where $\mathcal{T} = (t_0 = v_t, t_1, ..., t_k = v_t)$ denotes the truck tour with start and end node $t_0 = v_t$ and $\mathcal{B} = (b_0 = v_b, b_1, ..., b_l = v_b)$ the cargo bike tour. For the following notation, we assume that both tours $\mathcal{T}$, $\mathcal{B}$ are non-empty.

In particular, there must be at least one node $v \in V$ with $0 \neq d(v) \leq C_b$; otherwise, the bike cannot be used. In addition, to call $(\mathcal{T}, \mathcal{B})$ a feasible solution, further properties are required.

The combined tour $(\mathcal{T}, \mathcal{B})$ has to cover the demand of all nodes $v \in V$, i.e., each node has to be served by either the bike or the truck. Nodes visited by both vehicles, called combined nodes $(\kappa_1, ..., \kappa_m)$, are supplied by the truck and used to reload the cargo bike with the required goods. If $v_t \neq v_b$, the bike starts without any goods. Therefore, the first node visited has to be a combined node (i.e., $b_1 = \kappa_1$). Since the cargo bike has only a limited capacity, the summed demands of the nodes between two successive combined nodes on the bike tour $\mathcal{B}$ may not exceed C_b.

Problem 2 Let a complete digraph $G = (V, E)$ with edge costs $c^t(e), c^b(e)$, $e \in E$ for truck and cargo bike be given, as well as a bike capacity C_b and demand $d(v)$, $v \in V$, with no demand at the depots. The *combined truck-and-cargo-bike routing problem* (CTBRP) is to find a feasible combined tour $(\mathcal{T}, \mathcal{B})$ such that the generalized costs of the combined tour are minimized.

3.1 Cost structure

From the practical application, there are many different interpretations of the generalized costs that might be interesting to consider, especially concerning tour durations and distance covered. We, therefore, introduce various kinds of objective functions in Sect. 3.2. From a theoretical standpoint, the most important distinction is between *independent costs* and *synchronized costs*.

1. *Independent costs:* for some objectives, the costs of the truck tour and the bike tour can be computed separately, distributed to the edges as $c^t(e)$ and $c^b(e)$, respectively. This includes distance-based costs, i.e., a weighted sum of the distance covered by the truck and the cargo bike, but also emission-based costs.
2. *Synchronized costs:* when the duration of one or both tours is minimized, it does not suffice to model the objective independently. At each combined stop, the synchronization of both tours has to be guaranteed, i.e., the time since the last combined stop has to be long enough for both the truck and the cargo bike to serve all intermediate stops.

For independent costs, we define the costs of the truck and cargo bike tour as

$$c^t(\mathcal{T}) := \sum_{i=0}^{k-1} c^t(t_i, t_{i+1}) \quad \text{and} \quad c^b(\mathcal{B}) := \sum_{i=0}^{l-1} c^b(b_i, b_{i+1}).$$

For synchronized costs, we take into account that the vehicles have to wait for each other at the combined nodes. Subsequently, the duration between two successive combined nodes κ_i, κ_{i+1} on the corresponding tour is the costs of the slower vehicle (the vehicle with the higher summed costs) between the two nodes. To determine the costs of a tour, we need to sum up these durations between all successive combined nodes, as well as the summed costs (of the corresponding vehicle) between the last combined node κ_m and the depot. To describe this, we extend the notation and define the truck costs of a path from node t_i to node t_j $(i < j)$ regarding tour $\mathcal{T} = (t_0 = v_t, t_1, \dots, t_k = v_t)$ by

$$c^t(t_i, t_j, \mathcal{T}) := \sum_{x=i}^{j-1} c^t(t_x, t_{x+1}).$$

An equivalent definition applies to the bike costs regarding $\mathcal{B}, b_i$ and b_j ($i < j$):

$$c^b(b_i, b_j, \mathcal{B}) := \sum_{x=i}^{j-1} c^b(b_x, b_{x+1}).$$

Now we have

$$\hat{c}^t(\mathcal{T}) := \sum_{i=0}^{m-1} \max\left\{c^t(\kappa_i, \kappa_{i+1}, \mathcal{T}), c^b(\kappa_i, \kappa_{i+1}, \mathcal{B})\right\} + c^t(\kappa_m, v_t, \mathcal{T}), \tag{1}$$

and similarly, for

$$\hat{c}^b(\mathcal{B}) := \sum_{i=0}^{m-1} \max\left\{c^t(\kappa_i, \kappa_{i+1}, \mathcal{T}), c^b(\kappa_i, \kappa_{i+1}, \mathcal{B})\right\} + c^b(\kappa_m, v_b, \mathcal{B}). \tag{2}$$

We overload the notation and use $\kappa_0 := v_t$ in c^t and $\kappa_0 := v_b$ in c^b.

The preceding definition of the synchronized costs assumes that both vehicles start at the same time. This is useful in objective functions minimizing the delivery period, also known as completion time in the literature (e.g., Agatz et al. 2018). If we want to minimize the longest tour, the former formulation needs a slight modification. We can assume that the bike and the truck arrive at the first combined node κ_1 at the same time. Subsequently, we add the summed costs (of the corresponding vehicle) between the depot and the first combined node κ_1 instead of the maximum of both vehicles. This results in the following definition:

$$\tilde{c}^t(\mathcal{T}) := \sum_{i=1}^{m-1} \max\left\{c^t(\kappa_i, \kappa_{i+1}, \mathcal{T}), c^b(\kappa_i, \kappa_{i+1}, \mathcal{B})\right\} + c^t(v_t, \kappa_1, \mathcal{T}) + c^t(\kappa_m, v_t, \mathcal{T}), \tag{3}$$

and similarly, for

$$\tilde{c}^b(\mathcal{B}) := \sum_{i=1}^{m-1} \max\left\{c^t(\kappa_i, \kappa_{i+1}, \mathcal{T}), c^b(\kappa_i, \kappa_{i+1}, \mathcal{B})\right\} + c^b(v_b, \kappa_1, \mathcal{B}) + c^b(\kappa_m, v_b, \mathcal{B}). \tag{4}$$

Note that if $v_t \neq v_b$, the first node the cargo bike visits after the depot has to be a combined node. In this case, $c^b(v_b, \kappa_1, \mathcal{B}) = c^b(v_b, b_1) = c^b(v_b, \kappa_1)$ applies.

Obviously, $\hat{c}^t(\mathcal{T}) \geq \tilde{c}^t(\mathcal{T}) \geq c^t(\mathcal{T})$ and $\hat{c}^b(\mathcal{B}) \geq \tilde{c}^b(\mathcal{B}) \geq c^b(\mathcal{B})$ holds.

3.2 MIP formulations

In this section, we present our main MIP formulation for the combined truck-and-cargo-bike routing problem with one truck and one bike. This is based on synchronized (time-based) costs while the objective is to minimize the completion time (tbc_mct). Furthermore, we describe variations of the latter, taking into account different cost structures, objective functions and other relevant properties. Note that the MIP formulations only consider solutions $(\mathcal{T}, \mathcal{B})$ with $\mathcal{T} \neq \emptyset$ and $\mathcal{B} \neq \emptyset$. Therefore, an optimal solution of the

MIP formulation has to be compared to an optimal solution using only the truck or the bike, respectively, which can be computed as a TSP.

Starting with the main formulation (tbc_mct), we define binary variables $x^t_{(v,w)}$, $x^b_{(v,w)}$, x^t_v and x^b_v, which indicate, respectively, whether the corresponding edge $(v, w) \in E$ or node $v \in V$ is on the truck or bike tour.

The variables d_v represent the costs of the respective tour starting at v_t or v_b up to node $v \in V$, taking into account that both vehicles have to wait for each other at the combined nodes. More intuitively, d_v represents the time when the respective vehicle reaches node v or the time of the slower vehicle in case v is visited by both the truck and cargo bike. Regardless of the cost function, we need these variables to ensure that both vehicles visit the combined nodes in the same order.

Last, we define variables ℓ_v for $v \in V$ to take the limited bike capacity C_b into account. The value of ℓ_v describes the number of goods delivered from the bike up to and including node v, starting after the last reloading at a combined node. The resulting MIP is shown in (5)–(19).

Constraints (10)–(11) ensure that each truck and bike node has exactly one incoming and outgoing edge on the corresponding tour. With constraints (7), both tours start at their respective depot, and with (9), each node is supplied by at least one of the two vehicles. If $v_t \neq v_b$, constraints (12) ensure that the first node visited by the bike is a truck node; otherwise, (12) is not necessary.

$$\min \quad z \tag{5}$$

$$\text{s.t.} \quad d_{v_t}, d_{v_b} \leq z \tag{6}$$

$$x^t_{v_t}, x^b_{v_b} = 1 \tag{7}$$

$$x^b_{v_t}, x^t_{v_b} = 0 \tag{8}$$

$$1 \leq x^t_v + x^b_v \quad \forall\, v \in V \tag{9}$$

$$\sum_{\substack{w \in V \\ w \neq v}} x^t_{(v,w)} = \sum_{\substack{w \in V \\ w \neq v}} x^t_{(w,v)} = x^t_v \quad \forall\, v \in V \tag{10}$$

$$\sum_{\substack{w \in V \\ w \neq v}} x^b_{(v,w)} = \sum_{\substack{w \in V \\ w \neq v}} x^b_{(w,v)} = x^b_v \quad \forall\, v \in V \tag{11}$$

$$x^b_{(v_b,v)} \leq x^t_v \quad \begin{matrix} \forall\, v \in V \\ v \neq v_b \end{matrix} \tag{12}$$

$$\ell_v \leq (1 - x_v^t) \cdot C_b \quad \forall\, v \in V \tag{13}$$

$$\ell_v + d(w) - \ell_w \leq (1 - x_{(v,w)}^b + x_w^t) \cdot (C_b + \max_{v' \in V}\{d(v')\}) \quad \forall\, (v,w) \in E \tag{14}$$

$$x_{(v_t,v)}^t \cdot c^t(v_t, v) \leq d_v \quad \forall\, v \in V \tag{15}$$

$$x_{(v_b,v)}^b \cdot c^b(v_b, v) \leq d_v \quad \forall\, v \in V \tag{16}$$

$$c^t(v,w) \cdot x_{(v,w)}^t + c^b(v,w) \cdot x_{(v,w)}^b \leq d_w - d_v + (1 - x_{(v,w)}^t - x_{(v,w)}^b) \cdot M \quad \begin{array}{c}\forall\, (v,w) \in E \\ v \neq v_t, v_b\end{array} \tag{17}$$

$$x_e^t, x_e^b, x_v^t, x_v^b \in \{0, 1\} \quad \forall\, e \in E, v \in V \tag{18}$$

$$d_v, \ell_v \geq 0 \quad \forall\, v \in V \tag{19}$$

Lemma 3 *Constraints* (13)–(14) *ensure that the cargo bike visits a truck node for reloading* (*if necessary*) *without exceeding the capacity.*

Proof The value of ℓ_v describes the number of goods delivered from the bike up to and including node v, starting after the last reloading at a combined node. Consequently, this value may not exceed C_b and is set to 0 at each truck node, especially at all combined nodes (i.e., all combined nodes are served by the truck). Constraints (13) ensure the latter. If v is a truck node, it follows that $x_v^t = 1$, and therefore, $0 \leq \ell_v \leq 0$. Otherwise, we have $x_v^t = 0$ and $\ell_v \leq C_b$. If the bike drives from v to w (i.e., $x_{(v,w)}^b = 1$) and w is not a combined node (in particular, not a truck node, i.e., $x_w^t = 0$), it follows from constraints (14) that $\ell_v + d(w) \leq \ell_w$. For any other case, we get $(1 - x_{(v,w)}^b + x_w^t) \geq 1$, and since

$$\ell_v \leq C_b \text{ and } \quad d(w) \leq \max_{v \in V}\{d(v)\},$$

holds, we get

$$\underbrace{\ell_v}_{\leq C_b} + \underbrace{d(w)}_{\leq \max_{v \in V}\{d(v)\}} \leq \ell_w + \underbrace{(1 - x_{(v,w)}^b + x_w^t)}_{\geq 1} \cdot (C_b + \max_{v \in V}\{d(v)\})$$

and obtain $0 \leq \ell_w$. Consequently, there is no relevant bound on ℓ_w. □

Lemma 4 *The MIP formulation in* (5)–(19) *ensures that both vehicles meet at the same time at a combined node and takes the resulting waiting times into account.*

Proof The proof can be found in Appendix A. □

Note that in this formulation, both vehicles start at the same time. Consequently, objective (5) minimizes the completion time due to constraints (6).

Lemma 5 *Constraints* (17) *serve as subtour elimination constraints for the truck and bike tour*, *respectively*.

Proof The proof can be found in Appendix A. □

For all formulations, including synchronization between the vehicles, we need to choose M in constraints (17) large enough. In particular, M has to satisfy

$$d_v - d_w + \max_{(v,w)\in E}\{c^t(v,w), c^b(v,w)\} \leq \max_{v\in V}\{d_v\} + \max_{(v,w)\in E}\{c^t(v,w), c^b(v,w)\},$$

if the corresponding $x^t_{(v,w)}$ or $x^b_{(v,w)}$ is equal to zero.

While $\max_{(v,w)\in E}\{c^t(v,w), c^b(v,w)\}$ is easy to determine a priori, this is not readily possible for $\max_{v\in V}\{d_v\}$. Consequently, we have to estimate the latter by, for example, $\sum_{e\in E}\max\{c^t(e), c^b(e)\}$.

3.2.1 Time-based costs minimizing longest tour (tbc_mlt)

To assume the bike and the truck arrive at the first combined node κ_1 always at the same time, our MIP needs a slight modification. By adding continuous variables g^t and g^b, we indicate the gap between the departure time of the first and second vehicle. Subsequently, we have $g^t > 0, g^b = 0$ or $g^t = 0, g^b > 0$. In addition, we replace constraints (15), (16) and (6) by

$$d_{v_t} - g^t \leq z \tag{20}$$

$$d_{v_b} - g^b \leq z \tag{21}$$

$$x^t_{(v_t,v)} \cdot c^t(v_t, v) + g^t \leq d_v \quad \forall\, v \in V \tag{22}$$

$$x^b_{(v_b,v)} \cdot c^b(v_b, v) + g^b \leq d_v \quad \forall\, v \in V \tag{23}$$

$$g^t, g^b \geq 0. \tag{24}$$

Under this assumption, it follows that $d_{v_t} - g^t$ equals the truck tour costs in (3) and $d_{v_b} - g^b$ equals the bike tour costs in (4). Consequently, the objective function (5) minimizes the maximum of both tour costs due to constraints (20) and (21).

3.2.2 Time-based costs minimizing summed tour durations (tbc_mst)

To minimize the summed tour durations with vehicles starting at the same time, we can modify our MIP by removing constraints (6) and replacing the objective function (5) by

$$d_{v_t} + d_{v_b}. \tag{25}$$

Remark If vehicles do not have to start at the same time, we instead use the previous formulation (tbc_mlt), replace the objective (5) by

$$d_{v_t} + d_{v_b} - g^b - g^t, \tag{26}$$

and remove constraints (20) and (21).

3.2.3 Distance-based costs with synchronization (dbc_ws)

The calculation of the tour costs becomes much easier if $c^t(v, w)$ and $c^b(v, w)$ represent distance-based costs between node v and node w. As mentioned at the beginning of this section, we can easily sum up the costs of all edges used since there is no need to consider waiting times. Nevertheless, we cannot neglect the variables d_v and the corresponding constraints to ensure that both vehicles visit the combined nodes in the same order. Subsequently, we only remove (6) and replace the objective (5) in (5)–(19) by

$$\sum_{(v,w)\in E} \left(x^t_{(v,w)} \cdot c^t(v, w) + x^b_{(v,w)} \cdot c^b(v, w) \right). \tag{27}$$

3.2.4 Distance-based costs without synchronization (dbc_os)

By assuming that the truck can safely deposit the goods at the combined nodes until the bike arrives, the bike and truck do not need to be at a combined node at the same time to reload. Therefore, we can remove variables d_v and the associated constraints in the above formulation.

As a consequence, we have to add extra constraints to eliminate subtours in the bike and the truck tour. Those from the Miller–Trucker–Zemlin formulation in Miller et al. (1960) are suitable, as they are also compatible if we allow multiple bikes.

3.2.5 MIP model with multiple bikes

We can extend the previous MIP formulations by slightly modifying constraints (11), similar to the vehicle routing MIP formulation to model using multiple bikes.

Instead of one, we can set or limit the in- and outgoing edges from the bike depot to any constant number B^* of allowed cargo bikes. Constraints (28)–(29) represent the latter and replace constraints (11).

$$\sum_{\substack{w \in V \\ w \neq v}} x^b_{(v,w)} = \sum_{\substack{w \in V \\ w \neq v}} x^b_{(w,v)} = x^b_v \qquad \begin{matrix} \forall\, v \in V \\ v \neq v_b \end{matrix} \tag{28}$$

$$\sum_{\substack{w \in V \\ w \neq v_b}} x^b_{(v_b,w)} = \sum_{\substack{w \in V \\ w \neq v_b}} x^b_{(w,v_b)} \leq B^* \tag{29}$$

Note that by using more than one bike, d_{v_b} denotes the time when the last bike returns to the depot node. Therefore, we can only minimize the completion time or the distance-based versions since we cannot distinguish between different cargo bikes. Consequently, it is not possible to consider multiple bikes with different capacity restrictions in this model. To overcome this, we would have to use separate variables for each cargo bike.

3.2.6 Remarks and further variations

The different objective functions with the corresponding generalized costs are summarized in Table 2. As a reminder, $c^b(\mathcal{B})$ $(c^t(\mathcal{T}))$ correspond to the bike (truck) costs without waiting times, $\tilde{c}^b(\mathcal{B})$ $(\tilde{c}^t(\mathcal{T}))$ are the bike (truck) costs when both start independently, but wait for each other at the combined nodes and $\hat{c}^b(\mathcal{B})$ $(\hat{c}^t(\mathcal{T}))$ are the bike (truck) costs when both wait for each other at the combined nodes and start at the same time.

Although minimizing carbon emissions may seem reasonable, we have not included this objective function in Table 2. This is due to the fact that this leads to a minimization of the truck tour duration, which in most cases ends in a very long and expensive bike tour.

Further variations of the model, especially reformulations of models from Anderluh et al. (2017) and Murray and Chu (2015), can be found in Appendix B.3.

Moreover, in Appendix C, we discuss the relationship to the traveling salesperson problem as well as the capacitated vehicle routing problem and provide results on approximating the improvement of optimal solutions compared to TSP solutions.

4 Solution approaches

As CTBRP is NP-hard, we cannot expect to find good solutions by using standard MIP solvers for realistically sized instances, as can be seen in the experiments in Sect. 5. Therefore, we introduce different heuristic approaches here.

Table 2 Objectives and corresponding generalized costs

Name	Objective	Generalized costs	Cost type	MIP
(tbc_mct)	Completion time	$\max\{\hat{c}^t(\mathcal{T}), \hat{c}^b(\mathcal{B})\}$	Synchronized (time-based)	(5)–(19)
(tbc_mlt)	Longest tour	$\max\{\tilde{c}^t(\mathcal{T}), \tilde{c}^b(\mathcal{B})\}$	Synchronized (time-based)	(5), (7)–(14), (17)–(24)
(tbc_mst)	Summed tour durations	$\hat{c}^t(\mathcal{T}) + \hat{c}^b(\mathcal{B})$	Synchronized (time-based)	(7)–(19), (25)
(dbc_ws)	Total tour length	$c^t(\mathcal{T}) + c^b(\mathcal{B})$	Independent (distance-based)	(7)–(19), (27)
(dbc_os)	Total tour length	$c^t(\mathcal{T}) + c^b(\mathcal{B})$	Independent (distance-based)	(7)–(14), (18), (19), (27),[a]

[a] Subtour elimination constraints, e.g., those in Miller et al. (1960)

4.1 Clustering-based heuristics

We start with a simple heuristic by clustering the customers first and subsequently calculating combined tours in and between the clusters.

An easy first idea is to use a typical clustering algorithm, e.g., k-means clustering, based on the cost function to create a clustering of the customers. After that, we calculate an optimal combined tour in each cluster by ensuring that the number of customers in each cluster is sufficiently small. Finally, we link the local solutions through a truck and bike tour.

We will not give a more detailed description of this approach because of its obvious bottleneck: Good local solutions can lead to a bad global solution. The following example illustrates the latter.

Example 6 Consider the instance on the left side of Fig. 2. For x large enough, two clusters A and B result. Both have the same structure, as shown on the right side in Fig. 2.

In an optimal solution, the truck supplies the customers in cluster A and the bike those in cluster B (if $C_b = 3$, $v_D = v_t = v_b, 0 < \epsilon < 1$, and the demand is equal to 1 for all customers). However, since the local optimal solutions in the clusters use both the bike ($\mathcal{B}_A = (A_1, A_2, A_1), \mathcal{B}_B = (B_1, B_2, B_1)$) and the truck ($\mathcal{T}_A = (A_1, A_3, A_1), \mathcal{T}_B = (B_1, B_3, B_1)$), both clusters are served by both vehicles in the resulting global solution as well. For $\epsilon \longrightarrow 0$, the costs of this solution tend to infinity, but those of an optimal solution remain the same.

Thus, we propose an alternative clustering approach. After creating the clustering, we first calculate the combined tour between the clusters and, subsequently, the corresponding solutions in the clusters. To keep the size and number of clusters small enough, applying both previous steps recursively would be possible. We call this algorithm *Heuristic-Clustering* and give a more detailed description in Algorithm 1.

Algorithm 1 Heuristic-Clustering

```
$D \longleftarrow$ sum of all demands;
$C_b \longleftarrow$ bike capacity;
$\bar{k} \longleftarrow \lceil \frac{D}{C_b} \rceil$;
use k-means-algorithm to create clustering of all customers with $k := \bar{k}$ clusters;
replace each cluster $i$ with a dummy node $c_i$ corresponding to the cluster center $c_i$;
$N(c_i)$ denotes all nodes corresponding to the cluster with cluster center $c_i$;
assign summed demands of cluster nodes $N(c_i)$ to corresponding dummy node $c_i$;
use MIP (tbc_mct) with a time limit of 60 seconds to solve reduced instance;
if necessary, increase the time limit;
$b_1, ..., b_j \longleftarrow$ resulting bike nodes;                    // cluster centers
$t_1, ..., t_m \longleftarrow$ resulting truck nodes;                   // cluster centers
$\kappa_1, ..., \kappa_\ell \longleftarrow$ resulting combined nodes;     // cluster centers
for $i = 0$ to $\ell$ do
    // transform back to truck tour
    if $i = 0$ or $i = \ell$ then $\kappa_0, \kappa_{\ell+1} \longleftarrow v_t$ ;
    $T_i \longleftarrow \kappa_i \cup N(\kappa_{i+1}) \cup \bigcup_{t \in T} N(t)$, where $T$ corresponds to the truck nodes visited
      between $\kappa_i$ and $\kappa_{i+1}$;
    compute shortest Hamiltonian path in $T_i$ starting with $\kappa_i$ and ending at $\kappa_{i+1}$;
    add path to $\mathcal{T}$;
    // transform back to bike tour
    if $i = 0$ or $i = \ell$ then $\kappa_0, \kappa_{\ell+1} \longleftarrow v_b$ ;
    $B_i \longleftarrow \kappa_i \cup \kappa_{i+1} \cup \bigcup_{b \in B} N(b)$, where $B$ corresponds to the bike nodes visited
      between $\kappa_i$ and $\kappa_{i+1}$;
    compute shortest Hamiltonian path in $B_i$ starting with $\kappa_i$ and ending at $\kappa_{i+1}$;
    add path to $\mathcal{B}$;
end
compute solution value of combined tour $(\mathcal{T}, \mathcal{B})$
```

Note that a dummy node (corresponding to a cluster center) can only be served by the bike if the corresponding summed demands of the nodes contained in that cluster do not exceed the bike capacity. Depending on the clustering, certain clusters are thus assigned to the truck a priori. To compute the shortest Hamiltonian path between two combined nodes, we use simulated annealing.

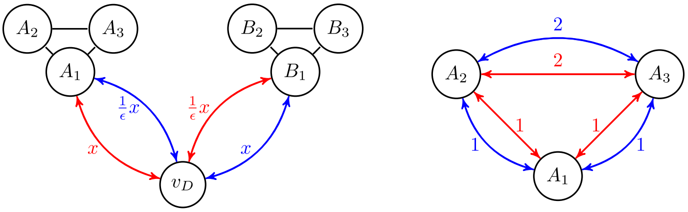


Fig. 2 Depot and two clusters (left) and the structure of a cluster on the right side with truck (red) and bike costs (blue) (color figure online)

4.2 TSP-based heuristic

Motivated by the results in Sect. C.2, we consider an algorithm that starts with the same tour for both vehicles (containing all nodes, e.g., an optimal TSP tour for the truck) and then successively deletes nodes in both tours to improve the solution. This is similar to the savings algorithm by Clarke and Wright (1964) but in a reversed fashion. Instead of merging tours until we obtain a feasible solution, we split both tours up while maintaining feasibility.

Algorithm 2 Heuristic-TSP

```
$\mathcal{T} \longleftarrow$ given truck tour;
$\mathcal{B} \longleftarrow \mathcal{T}$ but $v_t$ replaced with $v_b$;
$S \longleftarrow$ CalculateSavings($\mathcal{T}, \mathcal{B}$);   // nodes in non-increasing order of savings
while $S \neq \emptyset$ do
    $v_i$ first element in $S$;
    if $v_i$ truck node then  $\mathcal{T} \longleftarrow \mathcal{T} \setminus \{v_i\}$ ;
    if $v_i$ bike node then  $\mathcal{B} \longleftarrow \mathcal{B} \setminus \{v_i\}$ ;
    $S \longleftarrow$ CalculateSavings($\mathcal{T}, \mathcal{B}$);            // update saving list
end
```

Algorithm 3 CalculateSavings

```
Algorithm 3: CalculateSavings
for $t_i \in \mathcal{T}$ do
    if $(\mathcal{T} \setminus \{t_i\}, \mathcal{B})$ is feasible then
        $t_\alpha \longleftarrow$ preceding combined node;
        $t_\omega \longleftarrow$ succeeding combined node;
        $s(t_i) \longleftarrow \max\{c^t(t_\alpha, t_i, \mathcal{T}), c^b(t_\alpha, t_i, \mathcal{B})\} + \max\{c^t(t_i, t_\omega, \mathcal{T}), c^b(t_i, t_\omega, \mathcal{B})\} -$
          $\max\{c^t(t_\alpha, t_\omega, \mathcal{T} \setminus \{t_i\}), c^b(t_\alpha, t_\omega, \mathcal{B})\}$;            // savings
        $S \longleftarrow S \cup \{t_i\}$;    // including information about $s(t_i)$ and truck label
    end
end
for $b_i \in \mathcal{B}$ do
    if $(\mathcal{T}, \mathcal{B} \setminus \{b_i\})$ is feasible then
        $b_\alpha \longleftarrow$ preceding combined node;
        $b_\omega \longleftarrow$ succeeding combined node;
        $s(b_i) \longleftarrow \max\{c^t(b_\alpha, b_i, \mathcal{T}), c^b(b_\alpha, b_i, \mathcal{B})\} + \max\{c^t(b_i, b_\omega, \mathcal{T}), c^b(b_i, b_\omega, \mathcal{B})\} -$
          $\max\{c^t(b_\alpha, b_\omega, \mathcal{T}), c^b(b_\alpha, b_\omega, \mathcal{B} \setminus \{b_i\})\}$;            // savings
        $S \longleftarrow S \cup \{b_i\}$;    // including information about $s(b_i)$ and bike label
    end
end
sort all elements in $S$ in non-increasing order according to their savings $s(\cdot)$;
```

This algorithm, called *Heuristic-TSP*, is described in Algorithm 2, and the procedure how to select the nodes that are removed is described in Algorithm 3. The idea of the latter is to calculate the possible savings we get if we remove node v_i, which has to be a combined node since, in the other case, removing would not be feasible. If we delete this node in a tour, the corresponding vehicle can skip this node, and as a consequence, there is no need to wait at v_i for the respective other vehicle. The resulting savings are denoted in line 13 of Algorithm 3. If we consider a distance-based formulation instead of (tbc_mct), the resulting savings for truck node t_i can be calculated as follows:

$$s(t_i) := c^t(t_{i-1}, t_i) + c^t(t_i, t_{i+1}) - c^t(t_{i-1}, t_{i+1}), \tag{30}$$

and for bike node b_i

$$s(b_i) := c^b(b_{i-1}, b_i) + c^b(b_i, b_{i+1}) - c^b(b_{i-1}, b_{i+1}). \tag{31}$$

For the experimental evaluation in Sect. 5, we use simulated annealing to calculate the TSP tour for the truck and only consider the time-based variant.

4.3 Heuristic based on reinforcement learning

In this section, we take a slightly different approach using reinforcement learning. The idea of reinforcement learning is that one or more so-called agents interact with a dynamic environment and get instant feedback about their actions and the resulting changes to the environment. It belongs to the field of artificial intelligence and is used, among other things, in combinatorial optimization, e.g., for solving the TSP (Gambardella and Dorigo 1995; Júnior et al. 2010; Alipour and Razavi 2015; Zhang et al. 2020), which motivates its application in our model.

We adopt the reinforcement learning method *Q-learning* for our setting that is based on a Markov decision process (Buşoniu et al. 2010). In each state s, the agent has information about the environment and possible actions with the corresponding reward. Here, the Q-function $Q : S \times A \rightarrow \mathbb{R}$ approximates the expected return we get if we take the action $a \in A$ in state $s \in S$ and then follow an optimal policy. The latter means following the sequence of actions that leads to the maximal cumulative return. It follows that if the Q-function is optimal, we achieve an optimal policy by choosing in every state s an action a' that maximizes $Q(s, a')$.

To learn the Q-function, we start with an arbitrary one, and when we transit from state s to s' by taking action a and observing the reward r, we update $Q(s, a)$ as follows:

$$Q(s, a) \longleftarrow Q(s, a) + \alpha\Big(r + \gamma \max_{a' \in A} Q(s', a') - Q(s, a)\Big).$$

Here, α denotes the learning rate while the following term describes the difference between the current estimate $Q(s, a)$ and updated estimate $r + \gamma \max_{a' \in A} Q(s', a')$ of the Q-value. The second hyperparameter is the discount factor γ that allows modeling uncertainty about future rewards.

In our setting, the environment consists of the complete graph with nodes, edges, demands, positions and costs for both vehicles. We have an agent for the truck and one for the bike, respectively, with corresponding Q-functions, which we initialize with the negative costs of the respective cost function. The state s denotes the node that the corresponding vehicle is currently visiting (and the current load of the cargo bike), while the state space S includes all visitable nodes. In action space A, we have all nodes that can be served by the corresponding vehicle as the next node without violating any feasibility constraints. To exclude nodes in A that are already visited or to keep the bike from exceeding its capacity, we mask the corresponding actions, i.e., we temporarily set their Q-values to $-\infty$. Consequently, the next visited node corresponds to action a. The reward of the action a is equal to the negative time (costs) the current combined tour would take extra.

To have a balance between exploration and exploitation, we use the epsilon-greedy approach, i.e., in each step, we select a random next node with probability ϵ and otherwise, the action with the highest Q-value. Afterward, ϵ is updated by multiplying with $0 < \epsilon' < 1$. After arriving at the next state, we update the Q-function, the visited nodes, the positions of the vehicles and the current load of the bike.

To select the vehicle for the next step, we initialize the probability *prob* of choosing the truck as follows:

$$prob := \frac{\sum_{e \in E} c^t(e)}{\sum_{e \in E} c^t(e) + \sum_{e \in E} c^b(e)}.$$

Then, in each step, this value is updated in the following way

$$prob \longleftarrow 0.7 \cdot prob + 0.3 \frac{c^t(\mathcal{T})}{c^t(\mathcal{T}) + c^b(\mathcal{B})},$$

where $c^t(\mathcal{T})$ and $c^b(\mathcal{B})$ are the costs of the current truck and bike tour, respectively.

A training episode is finished after all nodes are visited and both vehicles are back at their corresponding depot. Our training consists of 1000 episodes, and we choose $\epsilon = 1, \epsilon' = 0.999, \gamma = 0.95$ and $\alpha = 0.8$.

5 Experimental evaluation

We evaluate six problem variants introduced in this paper on three classes of instances. The first class of artificial instances, $\mathcal{I}_1(n, \delta)$, see Fig. 3, is used to evaluate which parameters influence how difficult the problems are to solve. We especially evaluate the runtime and gap to an optimal solution for various settings of the number of customers n and speedup δ. More precisely, for a solution with value SOL and an optimal solution with value OPT, the gap refers to $\frac{SOL-OPT}{SOL}$.

The second and third classes of realistic instances, $\mathcal{I}_W(n, C_b)$ and $\mathcal{I}_M(n, C_b)$, see Fig. 7, consist of up to 250 addresses in Wuppertal and Münster, Germany, respectively. Here, we especially consider the solution quality compared to the TSP optimum.

The formulations with synchronized (time-based) costs are (tbc_mct), (tbc_mlt), (tbc_mst) with objective (25) and (tbc_mct2), which is the same formulation as (tbc_mct) but with up to two bikes as described in Sect. 3.2.5. Those with independent distance-based costs are (dbc_ws) and (dbc_os). For an overview of the objective functions, see Table 2. To solve the MIP formulations, we use Gurobi 8.1.1 (Gurobi 2019) and a time limit of 60 min for instances $\mathcal{I}_1(n, \delta)$ and 180 min for instances $\mathcal{I}_W(n, C_b)$ and $\mathcal{I}_M(n, C_b)$, respectively.

5.1 What makes the problems hard to solve?

Our first instance class $\mathcal{I}_1(n, \delta)$ consists of the example introduced in Lemma 11 in Appendix C.2, where we use six nodes, i.e., $n = 5$ customers. We extend this recursively by three nodes (i.e., $n \in \{5, 8, 11, 14, 17, 20, 23, 26, 29, 32, 35\}$) while we maintain the structure of the instance and vary the parameter $\delta \in [0.05, 1]$ in steps of 0.05. Recall that δ corresponds to the lowest speedup of the truck compared to the bike. In Fig. 3, instance $\mathcal{I}_1(8, \delta)$ is shown.

As described in Lemma 11, the structure of this instance class is theoretically perfectly suited to (tbc_mct) and (tbc_mlt). Additionally, we know that for both formulations, the optimal solution value for $\mathcal{I}_1(n, \delta)$ is $\frac{2}{3}\delta(1 + n)$.

Solving MIP formulations We first consider the influence of the parameters n and δ on the runtime of solving the MIP formulations, see Fig. 4 as well as Tables 3 and 4. Note that the runtime of the distance-based models (dbc_ws) and (dbc_os) is considerably lower than of the time-based models (tbc_mct) and (tbc_mlt). On average, the distance-based models can be solved about 5 times faster than the time-based ones (within a time limit of 60 minutes). This can be explained by the synchronization constraints (described in Lemma 4) and the fact that in the time-based models, we want to minimize tour durations, including waiting times, which are neglected in the distance-based formulations (especially in (dbc_os)). Tables 3 and 4 show that for the time-based models, the time limit of one hour leads to suboptimal solutions for $n \geq 11$. However, all distance-based models can be solved to optimality up to $n = 26$ and $n = 17$, respectively, within the time limit. Therefore, we consider the runtime of the distance-based models in Fig. 4a. As expected, increasing the number of demand points n leads to increased runtimes. However, also the speedup factor δ has a large influence on the runtime where instances with $\delta \in [0.15, 0.35]$ take considerably more time to solve. A similar correlation can be observed for the time-based models (tbc_mct) and (tbc_mlt) in Fig. 4b although the influence of the speedup δ is less pronounced. Note that for $n \geq 20$, the problem could not be solved to optimality for any δ, such that we report the gap of the best solution found within the time limit to the theoretically optimal solution $\delta\left(4 + \frac{2}{3}(n - 5)\right) = \frac{3}{2}\delta(1 + n)$, which can be derived from the example in Lemma 11. Note that this gap is considerably tighter than the MIP gap reported by Gurobi: For average n, the MIP gap ranged between 68 and 79%, and the gap to an optimal solution to (tbc_mct) and (tbc_mlt) varied from 25% to 55% depending on δ.

To further investigate the influence of the speedup factor δ on the runtime, we analyze the runtime of the models for varying values of δ averaged over all considered n in Fig. 5. As the runtime of the distance-based models is considerably shorter, we depict these separately in Fig. 5a. Here, we observe that adding

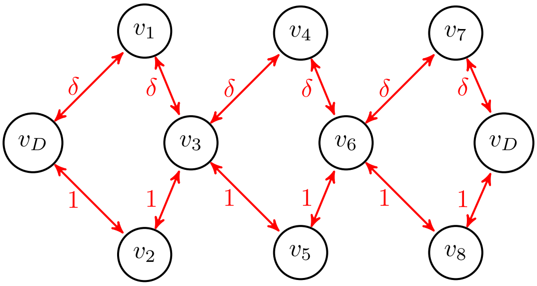


Fig. 3 Instance $\mathcal{I}_1(8, \delta)$ with corresponding edge costs of the truck, for the bike costs of edge $e \in E$ we have $c^b(e) := \delta \cdot c^t(e)$

synchronization constraints in (dbc_ws) significantly increases the runtime and that $\delta = 0.15$ results in the most difficult problems. When further increasing δ, the runtime reduces significantly.

For the time-based models, Fig. 5b shows a different correlation. For increasing δ, the average runtime of the time-based models increases. Only for $\delta = 1$, i.e., when truck and bike have the same speed, the runtime reduces again, possibly due to Lemma 14, which shows that for $\delta = 1$ the problem reduces to a TSP. This shows that adding the synchronized drive time and the waiting time into the objective structurally changes the problem and the solution process.

More detailed results can be found in Tables 3 and 4 in Appendix D.1.

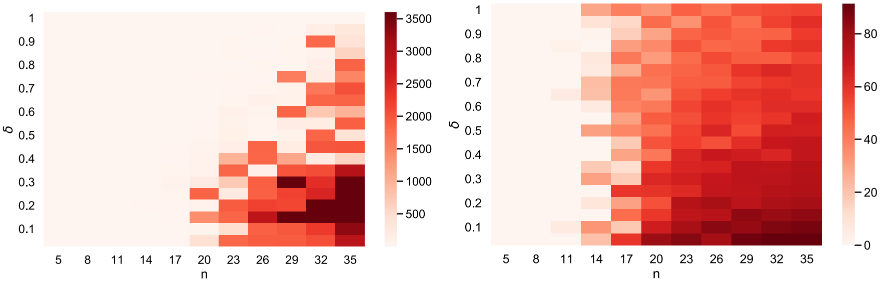
Heuristics In Fig. 6, we consider the influence of δ and n on the solution quality of for time-based model (tbc_mct) for *Heuristic-Clustering*, *Heuristic-TSP* and *Q-learning*. While for *Heuristic-TSP* and *Q-learning*, the influence of n seems to be marginal, the solution quality improves with increasing δ although the influence is rather small for $\delta > 0.4$. Similar behavior can be observed for *Heuristic-Clustering*, but here, the solution quality varies a lot with changing n. This is probably due to the low time limit of 60 s for solving the MIP, which is part of the solution process.

Averaged over all instances $\mathcal{I}_1(n, \delta)$, we observe a gap of 36, 44, 54 and 49% for the MIP, *Heuristic-TSP*, *Q-learning* and *Heuristic-Clustering*, respectively. If we exclude the five smallest instance sets, i.e., we only consider instances with $n \geq 20$; this leads to a gap of 59, 46, 60 and 70% for the previously mentioned solution methods. Thus, the heuristic solutions are competitive to MIP solutions for sufficiently large instances. This is especially relevant as overall, the solution times are considerably lower for the heuristics with $<$ 11 seconds for *Heuristic-TSP*, $<$ 60 seconds for *Q-learning* and $<$ 80 seconds for *Heuristic-Clustering*) compared to the runtimes of the MIP solver. While the runtime increases with increasing n, it appears to be to be independent of δ.

More detailed results can be found in Table 8 in Appendix D.1.

5.2 Improvements from TSP

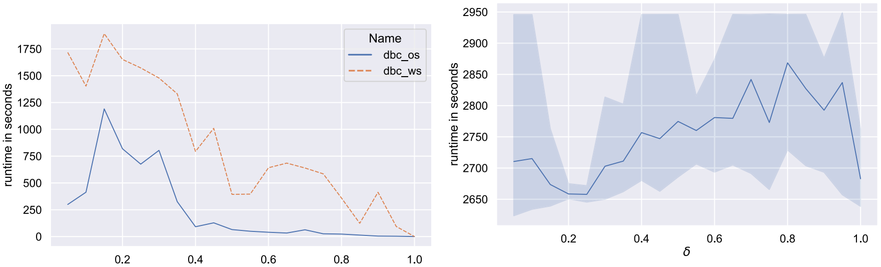
The second and third instance classes $\mathcal{I}_W(n, C_b)$ and $\mathcal{I}_M(n, C_b)$ consist of $n \in \{10, 20, 50, 100, 250\}$ addresses in Wuppertal (Fig. 7a) and Münster (Fig. 7b), respectively, with the corresponding distances and durations of the bike and



(a) Mean value of the runtime (in seconds) of the distance-based models.

(b) Mean value of the gap (in percent) of (tbc_mct) and (tbc_mlt).

Fig. 4 Instance $\mathcal{I}_1(n, \delta)$: Mean values of runtimes and gaps depending on the value of δ and number of customers n



(a) Runtime of the distance-based models (dbc_ws) and (dbc_os).

(b) Mean value and corresponding interval of the runtimes of the time-based models (tbc_mct), (tbc_mlt), (tbc_mst) and (tbc_mct2).

Fig. 5 Instance $\mathcal{I}_1(n, \delta)$: Runtimes depending on the value of δ and averaged over all n

Table 3 Results for algorithms (dbc_os), (dbc_ws) and (tbc_mst)

n	(dbc_os)			(dbc_ws)			(tbc_mst)		
	Runtime	MIP gap	Optimal	Runtime	MIP gap	Optimal	Runtime	MIP gap	Optimal
5	0.0	0.0	100.0	0.0	0.0	100.0	0.1	0.0	100.0
8	0.2	0.0	100.0	0.2	0.0	100.0	2.5	0.0	100.0
11	0.7	0.0	100.0	1.3	0.0	100.0	1159.4	0.6	95.0
14	2.0	0.0	100.0	6.5	0.0	100.0	3600.0	85.6	0.0
17	8.8	0.0	100.0	20.9	0.0	100.0	3600.0	92.1	0.0
20	23.0	0.0	100.0	414.4	0.4	95.0	3600.0	95.4	0.0
23	58.7	0.0	100.0	993.1	0.0	80.0	3600.0	97.2	0.0
26	215.5	0.0	100.0	1468.6	0.5	60.0	3600.0	97.9	0.0
29	531.9	1.3	90.0	1739.3	1.1	60.0	3600.0	98.1	0.0
32	726.6	1.0	90.0	2219.0	0.0	50.0	3600.0	98.4	0.0
35	1221.2	2.2	80.0	2587.1	0.0	40.0	3600.0	98.6	0.0

The percentage of instances solved to optimality for a given n is stated in column *optimal*. The gap to the optimal solution can be found in column *gap*. The runtime is given in seconds, and all other results are in percent

Table 4 Results for algorithms (tbc_mlt), (tbc_mct) and (tbc_mct2)

n	(tbc_mlt)				(tbc_mct)				(tbc_mct2)		
	Runtime	MIP gap	Gap	Optimal	Runtime	MIP gap	Gap	Optimal	Runtime	MIP gap	Optimal
5	0.1	0.0	0.0	100.0	0.1	0.0	0.0	100.0	0.1	0.0	100.0
8	9.6	0.0	0.0	100.0	2.8	0.0	0.0	100.0	2.3	0.0	100.0
11	2684.4	30.0	1.4	85.0	865.3	0.0	0.0	100.0	1186.8	0.9	95.0
14	3600.0	91.0	16.7	30.0	3600.0	86.9	7.7	55.0	3600.0	86.6	0.0
17	3600.0	94.7	33.0	5.0	3600.0	93.3	31.8	15.0	3600.0	90.4	0.0
20	3600.0	96.8	52.7	0.0	3600.0	95.7	43.6	0.0	3600.0	93.9	0.0
23	3600.0	98.0	61.1	0.0	3600.0	97.0	55.0	0.0	3600.0	96.7	0.0
26	3600.0	98.2	62.2	0.0	3600.0	97.8	60.6	0.0	3600.0	97.5	0.0
29	3600.0	98.7	63.6	0.0	3600.0	98.1	62.2	0.0	3600.0	98.0	0.0
32	3600.0	99.6	66.5	0.0	3600.0	98.4	66.0	0.0	3600.0	98.4	0.0
35	3600.0	100.0	68.2	0.0	3600.0	98.7	69.5	0.0	3600.0	98.7	0.0

The percentage of instances solved to optimality for a given *n* is stated in column *optimal*. The gap to the optimal solution can be found in column *gap*. The runtime is given in seconds, and all other results are in percent

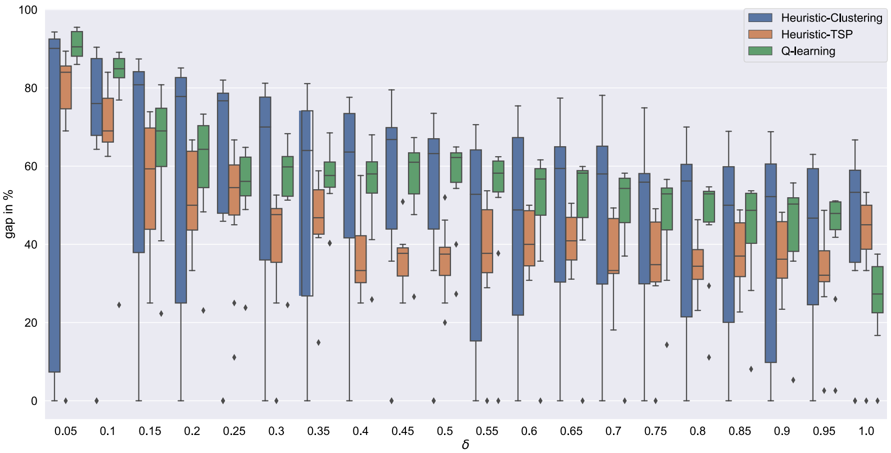


Fig. 6 Instance $\mathcal{I}_1(n, \delta)$: Gap values (in percent) of the algorithms depending on the value of δ for time-based model (tbc_mct). Each boxplot represents the variation of the gap depending on n. Here, the black line represents the median, and the box represents the 25th to 75th percentile of the gap values

truck. For each instance size, all associated locations are included in the next larger instance. We use the data provided by OpenStreetMap (Boeing 2017; OpenStreetMap contributors 2017) and typical speeds on the corresponding road types and vehicles. The bike depot is located in the inner city (red markers in Fig. 7), and the depot of the truck is in the northernmost and southernmost part of the map, respectively (blue markers in Fig. 7). All customers have a uniform demand equal to 1, while the capacity of the bike is equal to 50. Even though this is a common size in real-world applications, it is not reasonable to use such a high value for instances with less than 100 nodes. Therefore, we downsize the capacity for these instances, i.e., $C_b = 0.2n$. Additionally, we analyze the impact of varying the impact of the bike capacity.

With the selection of these instances, we pursue three main goals. The first is to compare cities with two different structures: While Münster is known to be very bike-friendly, Wuppertal is the opposite. In Table 5, the different values of δ and average ratios of the bike and truck weights underline the latter. The second goal is to investigate the influence of the bike capacity and compare the optimal solutions for the different objective functions. Finally, we study the behavior of our three algorithms with increasing instance size and compare it to the optimal TSP objective value. Note that in the TSP solution, the truck visits each node, including the bike and truck depot.

Comparing the solution structure between the models First, we consider the structure of solutions for the different time- and distance-based models for both Wuppertal ($\mathcal{I}_W(20, 4)$) and Münster ($\mathcal{I}_M(20, 4)$). Note that as in the previous case, the distance-based models are solved within a few seconds to optimality, while we obtain significantly higher runtimes for the distance-based models. Here, the MIP gap of (tbc_mct2) ranged between 17 and 36%, while all remaining formulations are solved to optimality.

Figure 8 shows the amount of time driving and waiting for both the truck and the bike, normalized by the duration of a TSP truck tour. As expected, there are considerable differences between the distance and time-based models, as well as between Wuppertal and Münster.

For the distance-based models, there are not sufficiently many shortcuts for the bike, but by construction, we have to use both vehicles, such that in both cases, the combined tour length is longer. The synchronization in (dbc_ws) additionally results in a high waiting time, such that the distance-based models provide no advantage compared to the TSP tour regarding the completion time.

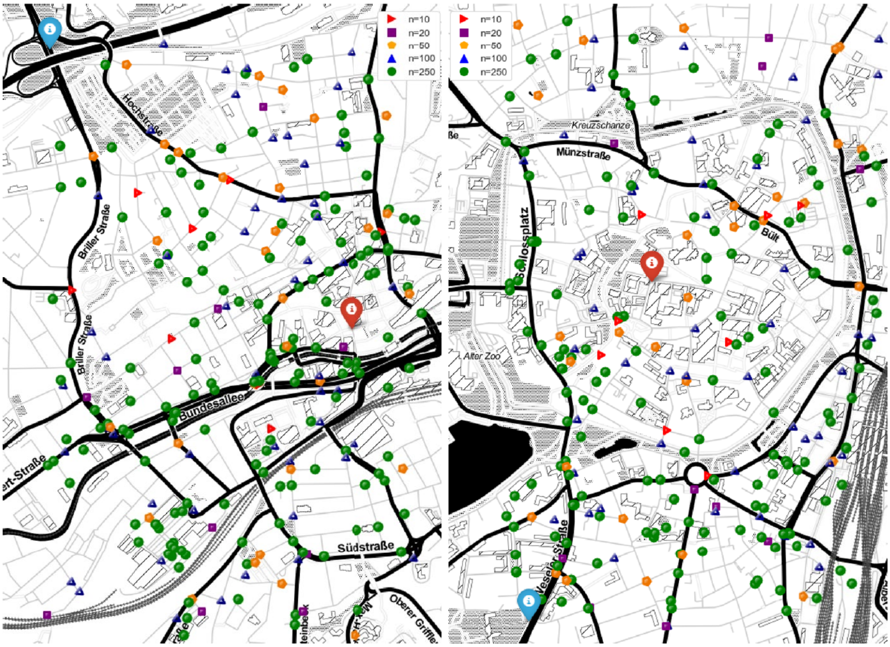
On the contrary, all time-based models reduce the duration of the truck tour significantly, and the respective completion time, as well as the longest tour duration, is lower than the duration of the TSP. Additionally, we observe that there is very little waiting time in the time-based models except for the model minimizing the completion time with two bikes, (tbc_mct2). Here, the sum of the driving times for both bikes is depicted such that the total driving time of the bike significantly exceeds that of one of the other models.

Note that for all models, both the duration of the truck tour and the completion time of the combined tour are lower for Münster than for Wuppertal. This can be attributed to the fact that for Münster, the mean speed difference between the bike and the truck is lower than for Wuppertal, see Table 5.

Influence of the bike capacity From an environmental perspective, it is most important to reduce the truck tour. In Fig. 9, we consider the influence of the bike capacity on reducing the truck tour duration for each model. For Wuppertal ($\mathcal{I}_W(20, C_b)$), Fig. 9a shows that depending on the model, the truck duration can be reduced by up to 46% where the time-based models have a significantly larger impact than the distance-based ones. However, increasing the bike capacity to more than 4, i.e., 20% of the truck capacity, has hardly any effect on the duration of the truck tour. For Münster ($\mathcal{I}_M(20, C_b)$), on the other hand, the reduction of the truck tour duration is more pronounced, as up to 51% of reduction are reached, see Fig. 9b. Here, the influence of increasing the bike capacity on reducing the truck tour duration is slightly larger, especially for the time-based model (tbc_mst) minimizing the summed tour duration.

Heuristic solutions for larger instances For both instances $\mathcal{I}_W$ and $\mathcal{I}_M$, the time limit of three hours allowed for solving instances of up to 20 demand points to optimality using the commercial MIP solver (Gurobi 2019). However, larger instances could not be solved optimally within this time frame, such that we consider the heuristics *Heuristic-Clustering, Heuristic-TSP* and *Q-learning* for solving (tbc_mct) as introduced in Sect. 4. Table 6 shows that even for $n = 250$, the runtimes of Heuristic-TSP and Heuristic-Clustering are below two minutes, while the runtime of Q-learning increases to up to 18 min.

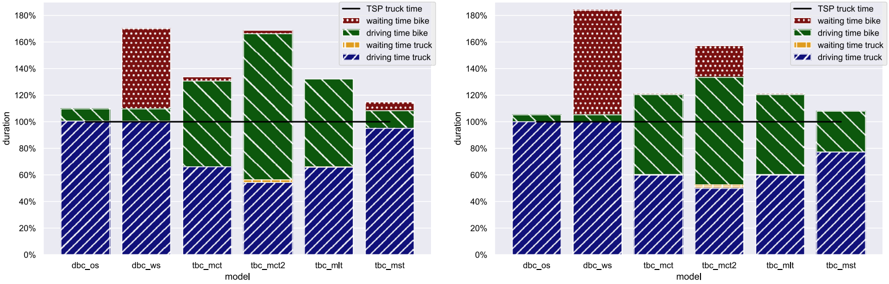
The difference between the two cities, Wuppertal and Münster, is also reflected in the solution quality of the heuristic solutions. Figure 10 depicts the objective value of (tbc_mct), i.e., the completion time of the heuristics solutions for a varying number of demand points. Here, the objective is normalized by the duration of a TSP truck tour for the same instance to facilitate comparing the solutions. First, note that for Wuppertal, see Fig. 10a, almost no solution improves upon



(a) Network of $\mathcal{I}_W(n, C_b)$.

(b) Network of $\mathcal{I}_M(n, C_b)$.

Fig. 7 $\mathcal{I}_W(n, C_b)$ and $\mathcal{I}_M(n, C_b)$. The red triangles correspond to $n = 10$ and the additional markers to the next larger instances: purple squares ($n = 20$), orange pentagons ($n = 50$), blue triangles ($n = 100$) and green circles ($n = 250$). The big red marker in the inner city represents the bike depot, and the blue one outside the city represents the truck depot (color figure online)

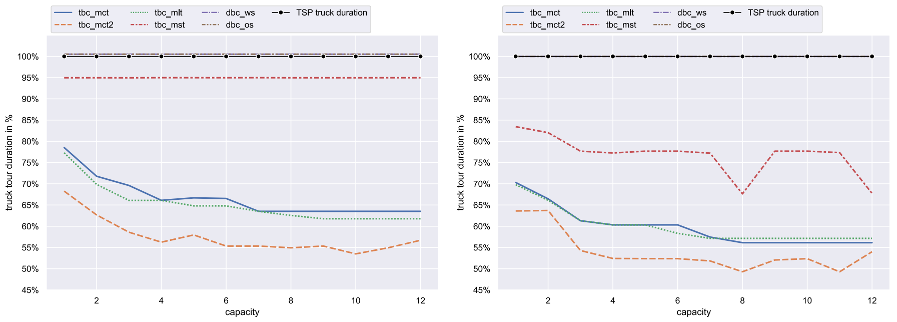


(a) $\mathcal{I}_W(20, 4)$.

(b) $\mathcal{I}_M(20, 4)$.

Fig. 8 Truck, bike and tour durations of different models with $C_b = 4$ for Wuppertal and Münster ($n = 20$)

the TSP tour. Only for 10 and 20 demand points, Heuristic-TSP and the MIP formulation find solutions that have a shorter completion time than the duration of the TSP tour. When comparing the heuristics to one another, it is apparent that Heuristic-Clustering performs worst while for larger instances with $n \geq 50$,



(a) $\mathcal{I}_W(20, C_b)$. (b) $\mathcal{I}_M(20, C_b)$.

Fig. 9 Truck tour duration depending on C_b and model, 100% corresponds to the duration of the truck TSP tour ($n = 20$)

Table 5 Average ratios of bike and truck weights and values of δ of the different instances $\mathcal{I}_W(n, C_b)$ and $\mathcal{I}_M(n, C_b)$

Instance	$n = 10$		$n = 20$		$n = 50$		$n = 100$		$n = 250$	
	W	M	W	M	W	M	W	M	W	M
$\delta := \min_{e\in E}\left\{\frac{c^b(e)}{c^t(e)}\right\}$	0.91	0.58	0.52	0.58	0.52	0.09	0.50	0.09	0.12	0.04
$\frac{1}{\lvert E\rvert}\sum_{e\in E}\frac{c^b(e)}{c^t(e)}$	1.52	1.13	1.51	1.26	1.52	1.25	1.50	1.23	1.49	1.17

Q-learning performs best. For small instances with $n \leq 20$, Heuristic-TSP performs best and is comparable to the optimal solution computed by a MIP solver as the gap is 9 and 20% for $n = 10$ and $n = 20$, respectively.

A similar behavior can be observed for Münster, see Fig. 10b, where Q-learning also finds the best solutions for large instances with $n \geq 100$ while Heuristic-Clustering performs worst. The most important difference, however, is that for all instances, the best heuristic solution outperforms the TSP solution. Even for $n = 250$, the completion time could be reduced to 75% of the TSP tour duration. Thus, the modeling approach developed in this paper helps to reduce the completion time of the delivery process and, consequently, the duration of truck tours.

All results can be found in Tables 9, 10 and 11 in Appendix D.2.

As we can see from the different results, Heuristic-Clustering has the advantage of being effective on small instance sizes, as it can often solve them exactly. However, due to the time limit and the exact solving process, which can become time-consuming, it performs worse on larger instances. Concerning the different graph structures, it is more suitable on instances with evenly distributed customers, as customers are closer together in the same cluster, allowing for more efficient bike routing. Also, Heuristic-TSP performs better on smaller instances since their optimal solutions of CTBRP share more similarities with the optimal TSP solutions, while on larger instance sizes, optimal solutions of CTBRP may deviate significantly from the TSP solution structurally. Q-learning consistently performs well across all instance sizes, contingent on the bike

Table 6 Runtimes of the different heuristic algorithms for solving (tbc_mct) on instances $\mathcal{I}_W(n, 0.2n)$ and $\mathcal{I}_M(n, 0.2n)$

	$n = 10$		$n = 20$		$n = 50$		$n = 100$		$n = 250$	
Instance	W	M	W	M	W	M	W	M	W	M
Heuristic-TSP	0	0	0	0	2	3	9	9	55	61
Heuristic-Clustering	3	2	2	2	5	9	18	220	102	73
Q-learning	2	3	10	11	36	39	158	198	908	1033

All times are given in seconds

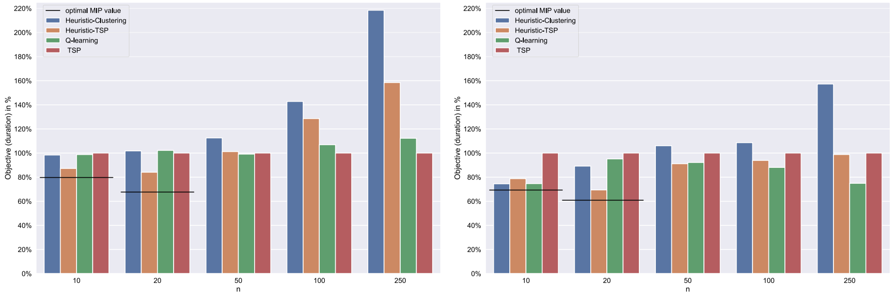
covering a larger number of customers without meeting the truck after each stop. This allows Q-learning to generate practical subtours for instance classes $\mathcal{I}_W(n, C_b)$ and $\mathcal{I}_M(n, C_b)$, as opposed to Heuristic-TSP, which is more suitable for $\mathcal{I}_1(n, \delta)$ where we have alternating bike and truck nodes.

6 Conclusion and future research

In this paper, we present a new concept for last-mile deliveries using two already established means of transport: the delivery truck and the cargo bike. We model this problem such that the truck both delivers goods and serves as a moving mini-depot for the cargo bike that has to be resupplied during the delivery route. By using the truck to restock the cargo bike, we do not need to construct mini-depots throughout the city. This makes our approach more flexible against changing demands and independent of available construction sites while also reducing investment costs. We develop a MIP formulation in various versions, either focusing on minimizing delivery time or distance, and extended this to include multiple cargo bikes. Moreover, we provide upper bounds compared to the truck-only delivery and theoretically analyze the complexity of the problem. To connect our approach to other delivery concepts from the literature, we show how our models can be adapted to cover these concepts as well.

In an experimental proof of concept, we analyze the performance of our different MIP formulations and show the advantages of our approach compared to truck-only delivery. These promising results show that we can outperform the traditional TSP approach in terms of completion time while reducing the distance driven by the truck and motivating further research in this field.

As a consequence of the limitations of MIP formulations in solving larger instances, we develop three heuristic solution approaches that can provide comparable solutions in a short amount of time. Due to the versatility of our model, covering a broad class of problems, those approaches might be useful to solve further problems in the literature, or conversely, we might adapt established methods. On the one hand, it would be interesting to consider the delivery problem as a multi-criteria problem where tour duration and distance covered are minimized simultaneously. On the other hand, uncertainties in the drive times have to be taken



(a) $\mathcal{I}_W(n, 0.2n)$.

(b) $\mathcal{I}_M(n, 0.2n)$.

Fig. 10 Completion time, i.e., objective value of (tbc_mct) in percent depending on number of demand points n and heuristic. 100% corresponds to the duration of the truck TSP tour of the corresponding instance size. Note that $C_b = 0.2n$

into account for making the model viable in practice. Here, it would be especially interesting to consider the case where truck and cargo bike are not affected in the same way by delays, e.g., when the bike can use a separate bike path and is not affected by traffic jams. Another interesting variant of the delivery problem is to integrate public transport planning, e.g., by using an already existing public transport line and fixing the corresponding nodes.

As part of future work, we plan to adapt deep Q-learning to learn the Q-values using a neural network and refine the reward function. Moreover, to reduce the complexity by one level, we plan to use a two-step approach where the assignment of stops to the truck and the cargo bike is fixed first and routes are constructed later.

A MIP formulation

Lemma 4 *The MIP formulation in* (5)–(19) *ensures that both vehicles meet at the same time at a combined node and takes the resulting waiting times into account.*

Proof Constraints (15) mark the time the first node v' on the truck tour is visited by setting $d_{v'} \geq c^t(v_t, v')$. In (16), the same applies for the first node v'' on the bike tour, and if $v' = v''$, it follows that $d_{v'} = \max\{c^t(v_t, v'), c^b(v_b, v')\}$.

Due to constraints (17), this process continues iteratively and guarantees the synchronization of the two vehicles at the combined nodes: If the truck drives from $v \neq v_t$ to w (i.e., $x^t_{(v,w)} = 1$), d_w is at least $d_v + c^t(v, w)$. Otherwise ($x^t_{(v,w)} = 0$), there is no bound on d_w by choosing M sufficiently large. We discuss the meaning of the latter in Sect. 3.2. The same reasoning applies to the bike tour since we can assume that both vehicles never use the same edge. □

Lemma 5 *Constraints* (17) *serve as subtour elimination constraints for the truck and bike tour, respectively.*

Proof These constraints are similar to those in the Miller–Trucker–Zemlin formulation of the TSP in Miller et al. (1960). They ensure that every truck tour contains the truck depot and every bike tour contains the bike depot. Otherwise, there would be at least one subtour that does not contain a depot node. Consequently, the values d_v, d_w of all consecutive nodes v and w along this subtour would have to satisfy the constraint

$$d_v + c^t(v, w) \leq d_w \quad \text{or} \quad d_v + c^b(v, w) \leq d_w,$$

depending on the subtour type. This is obviously not possible if $c^t(v, w) > 0$ or $c^b(v, w) > 0$ for at least one pair $(v,\ w)$. By merging nodes, if necessary, we can assume w.l.o.g. that $c^t(v, w) > 0$ and $c^b(v, w) > 0$ hold for all $(v, w) \in E$.

Since we only allow one outgoing and one incoming bike or truck edge at the associated depot in this formulation, it follows that there is exactly one bike and one truck tour. □

B Extending the model

B.1 Predefined nodes and satellite locations

As mentioned in Sect. 2, we can formulate the model in Anderluh et al. (2017) as a special case of our approach. Therefore, we set $x^t_v = 1$ and $x^b_v = 0$ ($x^t_v = 0$ and $x^b_v = 1$) for all predefined truck nodes (bike nodes) v and add a node with demand 0 for each satellite location. Constraints (9) can be removed. To permit multiple reload operations at the same satellite, we can add a sufficient number of node copies of each satellite. Consequently, all predefined truck nodes (bike nodes) are supplied exclusively by the truck (bike). If satellite s is used to reload the cargo bike, we obtain $x^t_s = 1$ and $x^b_s = 1$; otherwise, $x^t_s = 0$ and $x^b_s = 0$.

B.2 MIP model with cargo bike transportation

Since our approach is mainly motivated by Murray and Chu (2015), where a drone is used instead of a cargo bike that is able to park at the truck, we are interested in representing their model with our techniques.

So far, the cargo bike had to drive all roads by itself since all previous MIP formulations exclude free bike transportation by truck. This is due to the fact that we consider costs for both vehicles simultaneously. Consequently, in an optimal solution to the combined logistics problem, the bike and truck do not use the same edge $(a,\ b)$ in G. Even if they use the same edge starting at the same time, the faster vehicle would wait at the target node for the respective other one, and if we consider distance-based costs, both edge weights are added to the solution value. In both cases, the bike could skip node a and drive directly to b, and the triangle inequality ensures that the costs do not increase. As a consequence, it would only make sense to use the same edge if the truck could pick up the bike.

This leads us to the following modification and extension of our previous formulations. Therefore, we introduce a new variable y_e for each edge $e \in E$, which is equal to 1 if both vehicles use the corresponding edge e. To ensure this, we add the constraints (32)–(35):

$$y_e \leq x_e^t + x_e^b \quad \forall\, e \in E \tag{32}$$

$$y_e \leq 1 - x_e^t + x_e^b \quad \forall\, e \in E \tag{33}$$

$$y_e \leq 1 + x_e^t - x_e^b \quad \forall\, e \in E \tag{34}$$

$$y_e \in \{0, 1\} \quad \forall\, e \in E. \tag{35}$$

The resulting bounds on y_e are shown in Table 7. Instead of defining binary variables, we can replace (35) by $y_e \geq 0 \;\forall\, e \in E$.

As already mentioned, in an optimal solution, both vehicles would only drive along the same edge e ($y_e = 1$) if we benefit from transporting the cargo bike. Subsequently, the bike needs the same time $c^b(e) + y_e \cdot (c^t(e) - c^b(e)) = c^t(e)$ as the truck for driving along this edge e. Therefore, we replace constraints (16) by (36) and (37):

$$x_e^b \cdot c^b(e) + y_e \cdot (c^t(e) - c^b(e)) \leq d_v \quad \begin{array}{c} \forall\, v \in V \\ e = (v_b, v) \end{array} \tag{36}$$

$$d_v + c^b(e) + y_e \cdot (c^t(e) - c^b(e)) \leq d_w + (1 - x_e^b) \cdot M \quad \begin{array}{c} \forall\, e = (v, w) \in E \\ v \neq v_b \end{array}. \tag{37}$$

If necessary, we can provide a flight (bike) endurance limit C_f per subtour by adding a modified version of the capacity constraints:

$$f_v \leq (1 - t_v) \cdot C_f \quad \forall\, v \in V \tag{38}$$

$$f_v + c^b(v, w) - f_w \leq (1 - x^b_{(v,w)} + t_w) \cdot (C_f + \max_{e \in E}\{c^b(e)\}) \quad \forall\, (v, w) \in E \tag{39}$$

$$f_v \geq 0 \quad \forall\, v \in V. \tag{40}$$

Note that in our formulation, as well as in the model of Murray and Chu (2015), it is not possible for both vehicles to drive along an edge (or to visit a node) two times. While it would not make sense in the previous formulations, now it could be useful for the truck to drive along the same edge repeatedly. This is due to the fact that in some situations, the savings of picking up the cargo bike could be higher than the additional truck costs. To overcome this limitation, we can add the set V^N to V, which consists of N node copies $v^{(1)}, \ldots, v^{(N)}$ of each node $v \in V$ with demand 0 and

Table 7 Bounds (relevant ones in bold) on y_e for $e \in E$

x_e^t	x_e^b	(32)	(33)	(34)	(35)
0	0	$\leq \mathbf{0}$	≤ 1	≤ 1	$\geq \mathbf{0}$
0	1	≤ 1	≤ 2	$\leq \mathbf{0}$	$\geq \mathbf{0}$
1	0	≤ 1	$\leq \mathbf{0}$	≤ 2	$\geq \mathbf{0}$
1	1	≤ 2	$\leq \mathbf{1}$	≤ 1	$\geq \mathbf{0}$

$N := \left\lceil \frac{1}{C_b} \cdot \sum_{v \in V} d(v) \right\rceil - 1$. To ensure that it is not necessary to visit the node copies, we replace constraints (9) by

$$1 \leq t_v + b_v \quad \forall\, v \in V \setminus V^N.$$

In fact, this formulation also covers the model in Agatz et al. (2018).

B.3 Further variations

In all previous formulations, we use extra variables t_v and b_v to indicate whether node v is a truck or bike node. We can save those variables by replacing constraints (10) and (11) by

$$\sum_{\substack{w \in V \\ w \neq v}} x^t_{(v,w)} \leq 1 \quad \forall\, v \in V \tag{41}$$

and

$$\sum_{\substack{w \in V \\ w \neq v}} x^b_{(v,w)} \leq 1 \quad \forall\, v \in V \tag{42}$$

and all remaining t_v and b_v by $\sum_{\substack{w \in V \\ w \neq v}} x^t_{(v,w)}$ and $\sum_{\substack{w \in V \\ w \neq v}} x^b_{(v,w)}$, respectively.

Furthermore, time incurred at stop v due to delivery or searching for a parking space (depending on the stop, the vehicle and the demand $d(v)$) can be easily added to the edge costs $c^t(w, v)$ and $c^b(w, v)$ for all $v \in V$, $v \neq w$. This applies to all time-based costs.

By adding $\sum_{v \in V}(t_v + b_v - 1) \cdot c' + \sum_{v \in V} b_v \cdot d(v) \cdot c''$ to the objective, we can take into account the time c' needed for a reload operation and the loading time per package c''.

To include limitations of the cargo bike (truck) to visit node v, we can set $x^t_v = 1$ and $x^b_v = 0$ ($x^t_v = 0$ and $x^b_v = 1$).

If we want to use specified locations to reload the cargo bike, we can use the method in Appendix B.2. Thus, we add the set V^L, which contains these nodes (each with demand 0), to V and replace constraints (9) by

$$1 \leq t_v + b_v \quad \forall\, v \in V \setminus V^L.$$

C Theoretical properties

C.1 Relation to TSP and CVRP

As we can see, CTBRP generalizes several concepts from the literature that have been shown to be NP-hard. Thus, it is reasonable to expect that this applies to CTBRP as well.

Lemma 7 *CTBRP is a generalization of the traveling salesperson problem* (*TSP*) *and thus NP-hard* (*Laporte* 1992).

Proof Concerning distance-based costs, it is easy to see that the combined truck and cargo bike problem involves the TSP, which is NP-hard. By setting $c^b(e) := \infty\ \forall e \in E$, we would not use the bike, and the problem reduces to the asymmetric TSP. □

If we define $C_b := C_t$, $c^t \equiv c^b$, and $v_t = v_b$, both vehicles are equivalent in terms of costs, capacity and depot. However, (dbc_ws) does not reduce to CVRP with two vehicles, even when adding a capacity constraint to the truck, such as

$$\sum_{v \in V} x^t_v \cdot d(v) \leq C_t. \tag{43}$$

Lemma 8 *With the above assumptions, there are still feasible solutions for* (*dbc_ws*) *with lower costs than the corresponding optimal solution to CVRP.*

Proof An example is shown in Fig. 11. □

For $v_t := D =: v_b, C_b := 4 =: C_t$ and $\epsilon \in (0, 4]$ an optimal solution to (dbc_ws) would be $\mathcal{T} = \{D, t_1, \kappa_1, D\}$, $\mathcal{B} = \{D, b_1, \kappa_1, b_3, b_4, D\}$ but $\mathcal{T}_1 = \{D, t_1, b_1, \kappa_1, D\}$, $\mathcal{T}_2 = \{D, b_3, b_4, D\}$ to CVRP.

To create an alternative MIP formulation of CVRP with two vehicles, we additionally need to replace constraint (9) by

$$1 = x^t_v + x^b_v \quad \forall\, v \in V \tag{44}$$

or add a global capacity constraint for the cargo bike, such as

$$\sum_{v \in V} x_v^b \cdot d(v) \leq C_b. \tag{45}$$

For the latter case, constraints (13) and (14) are redundant.

When comparing (dbc_ws) and the MIP formulation of CVRP, the former produces on average 8% better solutions within the time limit of one hour. For this purpose, we use the CVRP instance sets B and P in Augerat (1995) with 33 instances ranging between 16 and 101 nodes. The capacities are scaled by a factor of $\frac{1}{2}$(original #vehicles), as we only allow two vehicles in all instances.

C.2 Approximating optimal solutions

In the following, we investigate how much better solutions of CTBRP can be compared to TSP and show that significant savings can be achieved. Note that we can interpret any feasible solution of the TSP as a feasible solution of CTBRP where only the truck is used.

In Sect. C.2.1, we analyze the time-based models (tbc_mct) and (tbc_mlt), while we give results for (tbc_mst) as well as the distance-based models (dbc_ws) and (dbc_os) in Sect. C.2.2.

C.2.1 Time-based costs minimizing completion time and the longest tour

In this section, we consider bounds on the value of an optimal solution to (tbc_mct) and (tbc_mlt). For simplicity, we assume $v_t = v_b$ and define $c(\mathcal{T}, \mathcal{B}) := \max\{c^t(\mathcal{T}), c^b(\mathcal{B})\}$ as the total costs of the combined tour, i.e., we neglect the waiting times of (tbc_mct) and (tbc_mlt), respectively. Furthermore, as described in Lemma 5, we can assume that $c^t(v, w) > 0$ and $c^b(v, w) > 0$ holds for all $(v, w) \in E$. Since

$$\max\{\hat{c}^t(\mathcal{T}), \hat{c}^b(\mathcal{B})\} \geq \max\{\tilde{c}^t(\mathcal{T}), \tilde{c}^b(\mathcal{B})\} \geq \max\{c^t(\mathcal{T}), c^b(\mathcal{B})\} \tag{46}$$

holds, we can use $c(\mathcal{T}, \mathcal{B}) := \max\{c^t(\mathcal{T}), c^b(\mathcal{B})\}$ in the following proofs, but the corresponding results also apply to (tbc_mct) (with $\max\{\hat{c}^t(\mathcal{T}), \hat{c}^b(\mathcal{B})\}$) and (tbc_mlt) (with $\max\{\tilde{c}^t(\mathcal{T}), \tilde{c}^b(\mathcal{B})\}$).

To evaluate both in relation to the case where we only use the truck, we set $\mathcal{TSP}^t$ as an optimal solution to the TSP regarding the cost function c^t, where $c^t(\mathcal{TSP}^t)$ denotes the summed truck costs of an optimal tour $\mathcal{TSP}^t$.

First, we consider the case where truck and cargo bike have the same speed and show that the solutions of CTBRP can be at most twice as good as optimal TSP solutions.

Lemma 9 *Let $(\mathcal{T}, \mathcal{B})$ be a feasible solution to an instance of CTBRP, and let $\mathcal{TSP}^t$ be an optimal solution to TSP for the same instance. Assuming $c^t(e) = c^b(e)$ for all $e \in E$, it holds that $c(\mathcal{T}, \mathcal{B}) \geq \frac{1}{2} \cdot c^t(\mathcal{TSP}^t)$.*

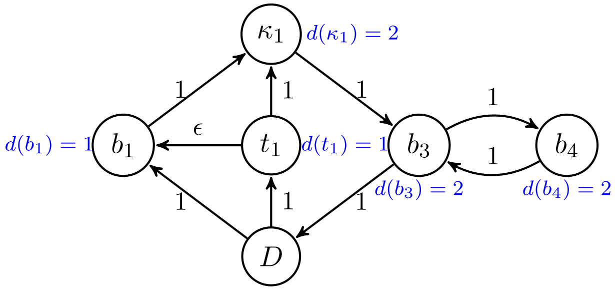


Fig. 11 Instance where the optimal solution value for (dbc_ws) and CVRP differs

Proof Since $v_t = v_b$, the cargo bike starts with the required goods from the depot, and thus, the first node b_1 visited by the bike is not a combined node. The same applies to the last node b_{l-1}, the bike visits before returning to the depot. Since it does not have to supply any nodes after b_{l-1}, there is no need to synchronize with the truck at b_{l-1}. Thus, we can assume that the last bike node is no combined node.

Now, if we ignore the capacity constraint of the cargo bike and skip all the combined nodes $\kappa_1, ..., \kappa_m$ on the bike tour, we get a tour $\mathcal{B}' := \mathcal{B} \backslash \{\kappa_1, ..., \kappa_m\}$ that has no node in common with $\mathcal{T}$ except for v_t. By the triangle inequality, the costs of this tour are at most as high as the costs of $\mathcal{B}$. In particular, there is no need for synchronization of bike and truck in $\mathcal{B}'$. So we have $c^t(\mathcal{B}) \geq c^t(\mathcal{B}')$.

Subsequently, we merge $\mathcal{T} = (v_t, t_1, ..., t_{k-1}, v_t)$ and $\mathcal{B}' = (v_t, b_1, ..., b_{l-1}, v_t) \backslash \{\kappa_1, ..., \kappa_m\}$ into one tour $\mathcal{T}' = (v_t ..., t_{k-1}, b_1, ..., v_t)$ by omitting v_t after t_{k-1} and before b_1 and connecting (t_{k-1}, b_1) directly. It holds that

$$c^t(\mathcal{T}) + c^t(\mathcal{B}') \geq c^t(\mathcal{T}'),$$

because $c^t(t_{k-1}, v_t) + c^t(v_t, b_1) \geq c^t(t_{k-1}, b_1)$ applies. Since the combined tour $(\mathcal{T}, \mathcal{B})$ covers all nodes and we only remove multiple nodes from both tours, it follows that $\mathcal{T}'$ also contains all nodes and forms a feasible solution for the TSP. In particular, $c^t(\mathcal{T}') \geq c^t(\mathcal{TSP}^t)$. Together, we have

$$2 \cdot \max\{c^t(\mathcal{T}), c^b(\mathcal{B})\}) \geq c^t(\mathcal{T}) + c^t(\mathcal{B}) \geq c^t(\mathcal{T}) + c^t(\mathcal{B}') \geq c^t(\mathcal{T}') \geq c^t(\mathcal{TSP}^t),$$

and thus $\dfrac{c^t(\mathcal{TSP}^t)}{c(\mathcal{T}, \mathcal{B})} \leq 2.$ □

In general, when comparing bike and truck weights for each edge in E, the minimum ratio $\min_{e \in E} \left\{ \frac{c^b(e)}{c^t(e)} \right\}$, i.e., the lowest speedup of the truck compared to the bike, can be easily determined. In the following, we denote this ratio by δ and show a generalized version of Lemma 9, inspired by the proof of Theorem 4.1 in Agatz et al. (2018).

Lemma 10 *Let $(\mathcal{T},\mathcal{B})$ be a feasible solution to an instance of CTBRP, and let $\mathcal{TSP}^t$ be an optimal solution to TSP for the same instance. If* $c^b(e) \geq \delta \cdot c^t(e)$for all $e \in E$, *it holds that*

$$c(\mathcal{T},\mathcal{B}) \geq \frac{\delta}{\delta+1} \cdot c^t(\mathcal{TSP}^t).$$

Proof Let $\lambda = \dfrac{c^t(\mathcal{T})}{c^t(\mathcal{TSP}^t)}$ and note that $\lambda > 0$. If $\lambda \geq 1$, the costs of the combined tour are higher than $c^t(\mathcal{TSP}^t)$ and the lemma follows immediately.

For $\lambda < 1$, we show that

$$c^b(\mathcal{B}) \geq (1-\lambda) \cdot \delta \cdot c^t(\mathcal{TSP}^t) \tag{47}$$

is satisfied. To prove this claim, we assume that the opposite holds. According to the assumption that $c^b(e) \geq \delta \cdot c^t(e)$ for all $e \in E$, it follows that $\delta \cdot c^t(\mathcal{B}) \leq c^b(\mathcal{B})$ and we have

$$c^t(\mathcal{B}) < (1-\lambda) \cdot c^t(\mathcal{TSP}^t). \tag{48}$$

Using the same procedure as in the proof of Lemma 9, we merge $\mathcal{T}$ and $\mathcal{B}$ into one tour $\mathcal{T}'$ and by the same argument,

$$c^t(\mathcal{TSP}^t) \leq c^t(\mathcal{T}') \leq c^t(\mathcal{T}) + c^t(\mathcal{B})$$

applies. However, from (48) we obtain

$$c^t(\mathcal{T}) + c^t(\mathcal{B}) < \lambda \cdot c^t(\mathcal{TSP}^t) + (1-\lambda) \cdot c^t(\mathcal{TSP}^t) = c^t(\mathcal{TSP}^t),$$

resulting in a contradiction and proving (47). Now, we can conclude

$$\begin{aligned} c(\mathcal{T},\mathcal{B}) &= \max\{c^t(\mathcal{T}), c^b(\mathcal{B})\} \geq \max\{\lambda \cdot c^t(\mathcal{TSP}^t), (1-\lambda) \cdot \delta \cdot c^t(\mathcal{TSP}^t)\} \\ &\geq \frac{\delta}{\delta+1} \cdot c^t(\mathcal{TSP}^t), \end{aligned}$$

where the last inequality holds since $\max\{\lambda \cdot c^t(\mathcal{TSP}^t), (1-\lambda) \cdot \delta \cdot c^t(\mathcal{TSP}^t)\}$ is minimal if $\lambda \cdot c^t(\mathcal{TSP}^t) = (1-\lambda) \cdot \delta \cdot c^t(\mathcal{TSP}^t)$. □

While the improvement of CTBRP compared to TSP is bounded by $\frac{\delta+1}{\delta}$, we show that this maximal improvement can actually be attained.

Lemma 11 *The upper bound in Lemma 10 respectively Lemma 9 is tight, i.e., there exists at least one instance for the above problem with*

$$\frac{c^t(\mathcal{TSP}^t)}{c(\mathcal{T},\mathcal{B})} = \frac{\delta+1}{\delta}.$$

Proof We show that the following instance $\mathcal{I}_1(5,\delta)$ has the required properties. Again, we have a complete symmetric digraph $G=(V,E)$ with $V = \{v_D, v_1, v_2, v_3, v_4, v_5\}$, $d(v_i) = 1$ for $i \in \{1,2,3,4,5\}$ and a cost function c^t

and c^b for the truck and the bike, respectively. The corresponding truck edge costs are defined as shown in Fig. 12a. For the bike costs of edge $e \in E$, we have $c^b(e) := \delta \cdot c^t(e)$. For reasons of clarity, we have drawn the depot twice, but it is the same node. Again, the costs of the undrawn edges are equal to the corresponding shortest paths. □

It is easy to verify that the value of an optimal solution to the MIP in (5)–(19) equals 4δ. A possible solution $(\mathcal{T}, \mathcal{B})$ is shown in Fig. 12c, while we have an optimal solution $\mathcal{TSP}^t$ to the TSP with cost function c^t and value $4 + 4\delta$ in Fig. 12b.

Subsequently, we obtain

$$\frac{c^t(\mathcal{TSP}^t)}{c(\mathcal{T}, \mathcal{B})} = \frac{4 + 4\delta}{4\delta} = \frac{\delta + 1}{\delta}.$$

We can use the bounds developed here to approximate the optimal objective value of CTBRP.

Corollary 12 *Let $\mathcal{TSP}^t$ be an optimal solution to the TSP. Then $(\mathcal{TSP}^t, \emptyset)$ is a $\frac{\delta+1}{\delta}$ -approximation to (tbc_mct) and (tbc_mlt).*

Proof Follows directly with Lemma 10 and (46). □

Corollary 13 *An α-approximation to the TSP is an $\alpha \cdot \frac{\delta+1}{\delta}$-approximation to (tbc_mct) and (tbc_mlt). Moreover, every α-approximation algorithm to the TSP is an $\alpha \cdot \frac{\delta+1}{\delta}$-approximation algorithm to (tbc_mct) and (tbc_mlt).*

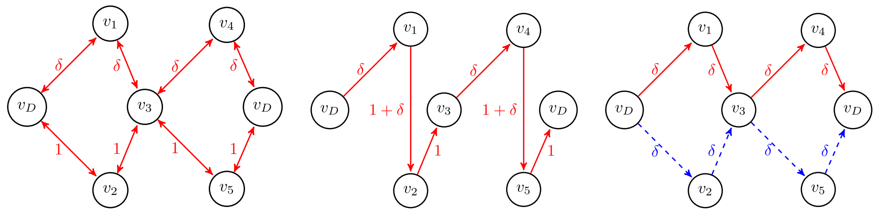
Proof Follows directly from Corollary 12. □

C.2.2 Time-based costs with summed tour durations and distance-based costs

When considering the summed tour durations or distance-based costs, the cargo bike must be faster than the truck on at least one edge to benefit from its use. Otherwise, it would only cause extra time or distance, respectively. This results in the following lemma.

Lemma 14 *Let $(\mathcal{T}, \mathcal{B})$ be an optimal solution to an instance of CTBRP, and let $\mathcal{TSP}^t$ be an optimal solution to TSP for the same instance. If $c^b(e) \geq c^t(e)$* for all *$e \in E$, it follows that $\hat{c}(\mathcal{T}, \mathcal{B}) := \hat{c}^t(\mathcal{T}) + \hat{c}^b(\mathcal{B}) = c^t(\mathcal{TSP}^t)$.*

Proof If $\mathcal{B} = \emptyset$, the lemma follows directly. Therefore, we assume $\mathcal{B} \neq \emptyset$. By neglecting the waiting times, the duration of both tours does not increase. Now, we proceed in a similar way as in the proof of Lemma 9. We skip the combined nodes on the bike tour and call this resulting bike tour $\mathcal{B}'$. Note that $\mathcal{B}'$ and $\mathcal{T}$ together still cover all nodes. Consequently, we have



(a) Edge costs of the truck, for the bike costs of edge $e \in E$ we have $c^b(e) := \delta \cdot c^t(e)$.

(b) An optimal solution to the truck-only variant.

(c) An optimal solution to the combined problem with truck (red) and bike tour (blue, dashed).

Fig. 12 Instance $\mathcal{I}_1(5,\delta)$ and optimal solutions to TSP and CTBRP

$$\hat{c}^t(\mathcal{T}) + \hat{c}^b(\mathcal{B}) \geq c^t(\mathcal{T}) + c^b(\mathcal{B}'),$$

since we only omit nodes and the triangle inequalities hold.

Now we merge $\mathcal{B}'$ and $\mathcal{T}$ into one tour $\mathcal{T}'$, forming a feasible solution of the TSP. With the same argument as above and since $c^b(e) \geq c^t(e)$ for all $e \in E$ hold, we have

$$c^t(\mathcal{T}) + c^b(\mathcal{B}') \geq c^t(\mathcal{T}') \geq c^t(\mathcal{TSP}^t). \tag{49}$$

As every feasible solution of the TSP is a feasible solution for our combined problem, the lemma follows. □

Corollary 15 *The result in Lemma 14 also holds for (dbc_ws) and (dbc_os).*

Proof Since we have $\hat{c}^t(\mathcal{T}) + \hat{c}^b(\mathcal{B}) \geq c^t(\mathcal{T}) + c^b(\mathcal{B})$, we can apply the same proof as in Lemma 14, except that we do not have waiting times. □

We can use these results to derive a lower bound. Therefore, we define $c''(e) := \min\{c^b(e), c^t(e)\}$ for all $e \in E$. Since the triangle inequalities need not hold for c'', we set $c'(e) := c''(sp''(e))$ for all $e \in E$. Here $sp''(e)$ denotes the shortest path from a to b (if $e = (a,b)$) regarding the cost function c''.

Lemma 16 *If $\mathcal{TSP}'$ represents an optimal solution to the TSP regarding the previously defined cost function c' and $(\mathcal{T},\mathcal{B})$ is a feasible solution of the same instance of CTBRP, then we have $\hat{c}(\mathcal{T},\mathcal{B}) = \hat{c}^t(\mathcal{T}) + \hat{c}^b(\mathcal{B}) \geq c'(\mathcal{TSP}')$.*

Proof The proof follows the one in Lemma 14 until we merge $\mathcal{B}'$ and $\mathcal{T}$ into one tour $\mathcal{T}'$. Since $c^t(\mathcal{T}) + c^b(\mathcal{B}') \geq c'(\mathcal{T}')$ holds, we can replace (49) by

$$c^t(\mathcal{T}) + c^b(\mathcal{B}') \geq c'(\mathcal{T}') \geq c'(\mathcal{TSP}')$$

without using the preconditions of Lemma 14. □

Corollary 17 *The result in Lemma 16 also holds for (dbc_ws) and (dbc_os).*

Thus, we can bound the solution quality of CTBRP from above and below by solving to instances of TSP.

D Experimental results

D.1 Instance class $\mathcal{I}_1(\boldsymbol{n}, \boldsymbol{\delta})$

Experimental results corresponding to Sect. 5.1 averaged over all δ. The percentage of instances solved to optimality for a given n is stated in column *optimal*. The gap to the optimal solution can be found in column *gap*. The runtime is given in seconds, and all other results are in percent (Table 8).

D.2 Instance class $\mathcal{I}_W(\boldsymbol{n}, \boldsymbol{C_b})$ and $\mathcal{I}_M(\boldsymbol{n}, \boldsymbol{C_b})$

Experimental results corresponding to Sect. 5.2. The instance classes $\mathcal{I}_W(n, C_b)$ and $\mathcal{I}_M(n, C_b)$ are denoted by W and M, respectively (Tables 9, 10 and 11).

Table 8 Results for algorithms Heuristic-Clustering, Heuristic-TSP and Q-learning in Fig. 6

n	Heuristic-Clustering			Heuristic-TSP			Q-learning		
	Runtime	Gap	Optimal	Runtime	Gap	Optimal	Runtime	Gap	Optimal
5	0.9	0.0	100.0	0.2	32.6	20.0	1.6	18.5	25.0
8	9.2	0.0	100.0	0.8	39.6	5.0	2.5	44.6	0.0
11	62.5	21.8	25.0	1.2	43.6	0.0	3.7	52.3	0.0
14	62.5	40.0	0.0	1.1	44.1	0.0	4.3	55.5	0.0
17	63.7	57.9	0.0	1.7	43.5	0.0	5.5	57.7	0.0
20	63.7	62.8	0.0	2.6	45.9	0.0	8.0	59.9	0.0
23	64.8	67.3	0.0	1.4	48.2	0.0	8.7	57.7	0.0
26	66.0	69.7	0.0	2.0	44.9	0.0	10.9	60.5	0.0
29	67.0	73.0	0.0	3.0	47.2	0.0	11.7	61.2	0.0
32	66.6	73.1	0.0	2.8	44.8	0.0	14.4	59.7	0.0
35	70.3	74.8	0.0	4.2	47.5	0.0	17.3	61.8	0.0

Table 9 Results for the different formulations in Fig. 8 with $C_b = 4$ and $n = 20$

Formulation	Waiting time bike		Driving time bike		Waiting time truck		Driving time truck	
	W	*M*	*W*	*M*	*W*	*M*	*W*	*M*
(tbc_mct)	4.7	1.2	98.8	85.0	0.0	0.7	101.2	84.7
(tbc_mct2)	3.9	33.8	168.2	114.6	2.9	3.7	83.2	70.5
(tbc_mlt)	0.0	1.2	101.1	85.0	0.5	0.7	100.6	84.7
(tbc_mst)	10.2	0.0	20.3	43.4	0.0	0.1	145.4	109.2
(dbc_ws)	92.4	111.4	14.0	7.3	0.0	0.0	153.8	141.6
(dbc_os)	0.0	0.0	14.0	7.3	0.0	0.0	153.8	141.6
TSP	0.0	0.0	0.0	0.0	0.0	0.0	153.0	141.6

Table 10 Results (truck tour durations) for the different formulations in Fig. 9 with $n = 20$

C_b	TSP		(dbc_os)		(dbc_ws)		(tbc_mct)		(tbc_mct2)		(tbc_mlt)		(tbc_mst)	
	W	*M*	*W*	*M*	*W*	*M*	*W*	*M*	*W*	*M*	*W*	*M*	*W*	*M*
1	153.0	141.6	153.8	141.6	153.8	141.6	120.2	99.5	104.5	90.0	118.3	98.8	145.3	118.1
2	153.0	141.6	153.8	141.6	153.8	141.6	109.8	94.1	95.9	90.2	106.9	93.6	145.3	116.1
3	153.0	141.6	153.8	141.6	153.8	141.6	106.5	86.8	89.7	76.8	101.1	86.8	145.3	110.0
4	153.0	141.6	153.8	141.6	153.8	141.6	101.2	85.4	86.1	74.2	101.1	85.4	145.4	109.4
5	153.0	141.6	153.8	141.6	153.8	141.6	102.0	85.4	88.7	74.1	99.2	85.4	145.4	110.0
6	153.0	141.6	153.8	141.6	153.8	141.6	101.8	85.4	84.7	70.4	99.2	82.6	145.3	110.0
7	153.0	141.6	153.8	141.6	153.8	141.6	97.2	81.3	84.7	73.4	97.2	80.9	145.4	109.3
8	153.0	141.6	153.8	141.6	153.8	141.6	97.2	79.4	84.0	69.8	95.7	80.9	145.3	95.7
9	153.0	141.6	153.8	141.6	153.8	141.6	97.2	79.4	84.7	73.7	94.5	80.9	145.3	110.0
10	153.0	141.6	153.8	141.6	153.8	141.6	97.2	79.4	81.9	74.1	94.5	80.9	145.3	110.0
11	153.0	141.6	153.8	141.6	153.8	141.6	97.2	79.4	84.0	69.7	94.5	80.9	145.3	109.5
12	153.0	141.6	153.8	141.6	153.8	141.6	97.2	79.4	86.8	74.1	94.5	80.9	145.3	95.9

Table 11 Results (completion time, i.e., objective value of (tbc_mct)) for the different heuristics with $C_b = 0.2n$ and TSP in Fig. 10

n	MIP		TSP		Heuristic-Clustering		Heuristic-TSP		Q-learning	
	W	*M*	*W*	*M*	*W*	*M*	*W*	*M*	*W*	*M*
10	60.5	63.7	75.9	92.0	74.7	68.5	66.2	72.4	75.0	68.6
20	103.5	86.2	153.0	141.6	155.8	126.2	128.7	98.2	156.4	134.7
50	–	–	260.7	270.7	293.4	287.2	263.9	246.8	258.4	249.6
100	–	–	345.9	437.4	493.9	475.0	444.8	410.6	369.8	385.2
250	–	–	629.1	1033.1	1374.2	1624.8	997.0	1020.4	706.3	773.7

Funding Open Access funding enabled and organized by Projekt DEAL. This work was partially supported by DFG under SCHO 1140/8-2 and by the Bundesministerium für Wirtschaft und Energie (BMWi) Grant Number 01MK20013A.

Availability of data and materials The datasets generated and analyzed during the experimental evaluation are available in Stinzendörfer and Schiewe (2024).

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

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

Alipour MM, Razavi SN (2015) A new multiagent reinforcement learning algorithm to solve the symmetric traveling salesman problem. Multiagent Grid Syst 11(2):107–119

Amorosi L, Puerto J, Valverde C (2021) Coordinating drones with mothership vehicles: the mothership and multiple drones routing problem with graphs. arXiv preprint arXiv:2109.01447

Anand N, Quak H, van Duin R, Tavasszy L (2012) City logistics modeling efforts: trends and gaps—a review. Procedia Soc Behav Sci 39:101–115

Anderluh A, Hemmelmayr VC, Nolz PC (2017) Synchronizing vans and cargo bikes in a city distribution network. CEJOR 25(2):345–376

Anderluh A, Nolz PC, Hemmelmayr VC, Crainic TG (2021) Multi-objective optimization of a two-echelon vehicle routing problem with vehicle synchronization and grey zone'customers arising in urban logistics. Eur J Oper Res 289(3):940–958

Applegate DL, Bixby RE, Chvátal V, Cook WJ (2011) The traveling salesman problem: a computational study. Princeton University Press

Augerat P (1995) Approche polyèdrale du problème de tournées de véhicules. PhD thesis, Institut National Polytechnique de Grenoble-INPG

Baldacci R, Battarra M, Vigo D (2008) Routing a heterogeneous fleet of vehicles. The vehicle routing problem: latest advances and new challenges. Springer, US, Boston, MA, pp 3–27

Bektas T, Crainic TG, Van Woensel T (2015) From managing urban freight to smart city logistics networks, Project report

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

Bertsimas DJ (1992) A vehicle routing problem with stochastic demand. Oper Res 40(3):574–585

Boeing G (2017) Osmnx: new methods for acquiring, constructing, analyzing, and visualizing complex street networks. Comput Environ Urban Syst 65:126–139

Bogyrbayeva A, Yoon T, Ko H, Lim S, Yun H, Kwon C (2023) A deep reinforcement learning approach for solving the traveling salesman problem with drone. Transp Res Part C: Emerg Technol 148:103981

Bouman P, Agatz N, Schmidt M (2018) Dynamic programming approaches for the traveling salesman problem with drone. Networks 72(4):528–542

Boysen N, Fedtke S, Schwerdfeger S (2021) Last-mile delivery concepts: a survey from an operational research perspective. OR Spectrum 43:1–58

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

Buşoniu L, Babuška R, Schutter BD (2010) Multi-agent reinforcement learning: an overview. Innov Multi-agent Syst Appl 1:183–221

Chao I-M (2002) A tabu search method for the truck and trailer routing problem. Comput Oper Res 29(1):33–51

Clarke G, Wright JW (1964) Scheduling of vehicles from a central depot to a number of delivery points. Oper Res 12(4):568–581

Contardo C, Hemmelmayr V, Crainic TG (2012) Lower and upper bounds for the two-echelon capacitated location-routing problem. Comput Oper Res 39(12):3185–3199

Crişan GC, Nechita E (2019) On a cooperative truck-and-drone delivery system. Proced Comput Sci 159:38–47

Cuda R, Guastaroba G, Speranza MG (2015) A survey on two-echelon routing problems. Comput Oper Res 55:185–199

Desrochers M, Desrosiers J, Solomon M (1992) A new optimization algorithm for the vehicle routing problem with time windows. Oper Res 40(2):342–354

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

Drexl M (2013) Applications of the vehicle routing problem with trailers and transshipments. Eur J Oper Res 227(2):275–283

Dror M, Trudeau P (1990) Split delivery routing. Naval Res Log (NRL) 37(3):383–402

Gambardella LM, Dorigo M (1995) Ant-q: a reinforcement learning approach to the traveling salesman problem. In: Machine learning proceedings. Elsevier, pp 252–260

Gendreau M, Laporte G, Séguin R (1995) An exact algorithm for the vehicle routing problem with stochastic demands and customers. Transp Sci 29(2):143–155

Ghilas V, Demir E, Van Woensel T (2016) The pickup and delivery problem with time windows and scheduled lines. INFOR: Inf Syst Oper Res 54(2):147–167

Goetschalckx M, Jacobs-Blecha C (1989) The vehicle routing problem with backhauls. Eur J Oper Res 42(1):39–51

Gonzalez-Feliu J, Perboli G, Tadei R, Vigo D (2008) The two-echelon capacitated vehicle routing problem, working paper or preprint

Grangier P, Gendreau M, Lehuédé F, Rousseau L-M (2016) An adaptive large neighborhood search for the two-echelon multiple-trip vehicle routing problem with satellite synchronization. Eur J Oper Res 254(1):80–91

Gurobi Optimizer (2019) Gurobi optimizer version 8.1.1. Gurobi Optimization, Inc., Houston. http://www.gurobi.com/

Hemmelmayr VC, Cordeau J-F, Crainic TG (2012) An adaptive large neighborhood search heuristic for two-echelon vehicle routing problems arising in city logistics. Comput Oper Res 39(12):3215–3228

Jünger M, Reinelt G, Rinaldi G (1995) Chapter 4 the traveling salesman problem. In: Network models, volume 7 of handbooks in operations research and management science. Elsevier, pp 225–330

Júnior FCDL, Neto ADD, De Melo JD (2010) Hybrid metaheuristics using reinforcement learning applied to salesman traveling problem. In: Traveling salesman problem, theory and applications. IntechOpen

Laporte G (1992) The traveling salesman problem: an overview of exact and approximate algorithms. Eur J Oper Res 59(2):231–247

Laporte G, Louveaux F, Mercure H (1992) The vehicle routing problem with stochastic travel times. Transp Sci 26(3):161–170

Li H, Zhao J, Zhan Z (2022) Truck and unmanned vehicle routing problem with time windows: a satellite synchronization perspective. J Adv Transport

Li H, Wang H, Chen J, Bai M (2021) Two-echelon vehicle routing problem with satellite bi-synchronization. Eur J Oper Res 288(3):775–793

Lin S-W, Vincent FY, Lu C-C (2011) A simulated annealing heuristic for the truck and trailer routing problem with time windows. Expert Syst Appl 38(12):15244–15252

Liu Y, Liu Z, Shi J, Wu G, Pedrycz W (2020) Two-echelon routing problem for parcel delivery by cooperated truck and drone. IEEE Trans Syst, Man, Cybern: Syst 51(12):7450–7465

Malandraki C, Daskin MS (1992) Time dependent vehicle routing problems: formulations, properties and heuristic algorithms. Transp Sci 26(3):185–200

Miller CE, Tucker AW, Zemlin RA (1960) Integer programming formulation of traveling salesman problems. J ACM 7(4):326–329

Murray CC, Chu AG (2015) The flying sidekick traveling salesman problem: optimization of drone-assisted parcel delivery. Transp Res Part C: Emerg Technol 54:86–109

Nguyen MA, Hà MH (2023) The parallel drone scheduling traveling salesman problem with collective drones. Transport Sci

Nguyen V-P, Prins C, Prodhon C (2012) Solving the two-echelon location routing problem by a grasp reinforced by a learning process and path relinking. Eur J Oper Res 216(1):113–126

OpenStreetMap contributors (2017) Planet dump retrieved from https://www.openstreetmap.org

Perboli G, Tadei R, Vigo D (2011) The two-echelon capacitated vehicle routing problem: models and math-based heuristics. Transp Sci 45(3):364–380

Russo F, Comi A (2020) Investigating the effects of city logistics measures on the economy of the city. Sustainability 12(4):1439

Schermer D, Moeini M, Wendt O (2020) A b ranch-and-cut approach and alternative formulations for the traveling salesman problem with drone. Networks 76(2):164–186

Sluijk N, Florio AM, Kinable J, Dellaert N, Van Woensel T (2023) Two-echelon vehicle routing problems: a literature review. Eur J Oper Res 304(3):865–886

Stinzendörfer M, Schiewe P (2024) Supplementary material for publication “optimizing combined tours—the truck-and-cargo-bike case”. https://doi.org/10.5281/zenodo.10591544

Sutrisno H, Yang C-L (2023) A two-echelon location routing problem with mobile satellites for last-mile delivery: mathematical formulation and clustering-based heuristic method. Ann Oper Res 323(1–2):203–228

Taniguchi E, Thompson RG, Qureshi AG (2020) Modelling city logistics using recent innovative technologies. Transp Res Proced 46:3–12

Tong B, Wang J, Wang X, Zhou F, Mao X, Zheng W (2022) Optimal route planning for truck-drone delivery using variable neighborhood tabu search algorithm. Appl Sci 12(1):529

Toth P, Vigo D (2014) Vehicle routing: problems, methods, and applications. SIAM

Vu L, Vu DM, Hà MH, Nguyen V-P (2022) The two-echelon routing problem with truck and drones. Int Trans Oper Res 29(5):2968–2994

Wen M, Larsen J, Clausen J, Cordeau J-F, Laporte G (2009) Vehicle routing with cross-docking. J Oper Res Soc 60:1708–1718

Zäpfel G, Bögl M (2008) Multi-period vehicle routing and crew scheduling with outsourcing options. Int J Prod Econ 113(2):980–996

Zhang R, Prokhorchuk A, Dauwels J (2020) Deep reinforcement learning for traveling salesman problem with time windows and rejections. In: 2020 international joint conference on neural networks (IJCNN). IEEE, pp 1–8

Zhou H, Qin H, Cheng C, Rousseau L-M (2023) An exact algorithm for the two-echelon vehicle routing problem with drones. Transp Res Part B: Methodol 168:124–150

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.