EURO Journal on Transportation and Logistics 14 (2025) 100159

Contents lists available at ScienceDirect

EURO Journal on Transportation and Logistics

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

A general modeling and simulation framework for dynamic vehicle routing

Markó Horváth [a,*], Tímea Tamási [b,a]

[a] HUN-REN Institute for Computer Science and Control, H-1111 Budapest, Kende u. 13-17, Hungary
[b] Department of Operations Research, Institute of Mathematics, ELTE Eötvös Loránd University, Budapest, Hungary

ARTICLE INFO

Keywords:
Dynamic vehicle routing
Modeling framework
Simulation framework
Discrete-event based decision process

ABSTRACT

In dynamic vehicle routing problems (DVRPs), some part of the information is revealed or changed on the fly, and the decision maker has the opportunity to re-plan the vehicle routes during their execution, reflecting on the changes. Accordingly, the solution to a DVRP is a flexible policy rather than a set of fixed routes. A policy is a problem-specific algorithm that is invoked at various decision points in the planning horizon and returns a decision according to the current state. Since DVRPs involve dynamic decision making, a simulator is an essential tool for dynamically testing and evaluating the policies. Despite this, there are few tools available that are specifically designed for this purpose. To fill this gap, we have developed a simulation framework that is suitable for a wide range of dynamic vehicle routing problems and allows to dynamically test different policies for the given problem. In this paper, we present the background of this simulation tool, for which we proposed a general modeling framework suitable for formalizing DVRPs independently of simulation purposes. Our open source simulation tool is already available, easy to use, and easily customizable, making it a useful tool for the research community.

1. Introduction

A vehicle routing problem is *dynamic*, if some part of the information is revealed or changed on the fly, and the decision maker (the service provider) has the opportunity to re-plan the vehicle routes during their execution, reflecting on the changes. Dynamic vehicle routing problems (DVRPs) have received a lot of attention in the past decades, which is certified by a series of recent review papers, e.g., Berbeglia et al. (2010), Pillac et al. (2013), Bektaş et al. (2014), Psaraftis et al. (2016), Ritzinger et al. (2016), Rios et al. (2021), Soeffker et al. (2022), Zhang and Van Woensel (2023), Mardešić et al. (2023). This growing interest is due to the wide range of real-world applications and the fact that today's technology enables real-time decision making.

Nowadays, DVRPs are usually modeled using the so-called *sequential decision process* (e.g., Ulmer et al. 2020, Soeffker et al. 2022). Briefly stated, the decision process transitions from decision point to decision point, where the decision maker is provided with the current state (i.e., all the available information) and has the opportunity to make a decision (e.g., update the vehicle routes), or in other words, to choose an action, see Fig. 1(a). Accordingly, a solution to the dynamic problem is a *policy*, which is a function that assigns a decision to every state.

Apart from survey articles, in the majority of the papers dealing with DVRPs, the authors propose policies for the problem at hand, and perform computational experiments to evaluate them, e.g., to compare them with state-of-the-art or baseline policies. In addition to doing the obviously necessary implementation of their policy, they need some kind of simulator for dynamic evaluation. In this paper, we focus on this dynamic evaluation, and we approach the DVRPs from the simulation point of view. Even more emphasized, our focus is not on the solution approaches for a particular DVRP, but on the modeling of general problems and on the dynamic testing of arbitrary solution methods.

According to our primary goal, we have implemented a simulation framework that is suitable for a wide range of dynamic vehicle routing problems and allows to dynamically test different solution approaches for the modeled problem. This article, however, is much more than technical documentation, as we also propose a general modeling framework suitable for formalizing DVRPs independently of simulation purposes.

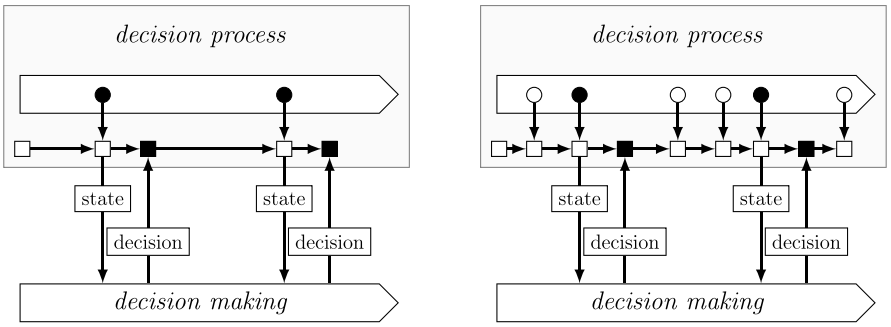
1.1. Motivation

The simulation of the decision process is essential for the dynamic evaluation of solution approaches to dynamic vehicle routing problems. Despite this, there are few tools available that are *specifically* designed for this purpose.

In simpler cases, it is very easy to implement the sequential decision process, since the transition between the states is straightforward.

* Corresponding author.
E-mail addresses: marko.horvath@sztaki.hu (M. Horváth), tamasitimea@student.elte.hu (T. Tamási).

https://doi.org/10.1016/j.ejtl.2025.100159
Received 19 November 2024; Received in revised form 20 March 2025; Accepted 24 April 2025
Available online 9 May 2025
2192-4376/© 2025 The Authors. Published by Elsevier B.V. on behalf of Association of European Operational Research Societies (EURO). This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).



(a) Sequential decision process. (b) Discrete-event based decision process.

Fig. 1. Differences between the sequential and the discrete-event based decision process. Circles refer to distinct events (e.g., order requests, vehicle arrival). Black circles refer to decision points. Squares refer to states. Black squares refer to post-decision states.

However, in many other cases (especially when inter-route constraints make the problem difficult), it is necessary to run a more complex simulation to move the decision process from decision point to decision point. Although general-purpose simulation tools exist (e.g., AnyLogic, SimPy), they require the user to build the entire dynamic vehicle routing framework from scratch. Several publicly available simulators have been created using these tools, but they are only suitable for a specific problem (see e.g., Hao et al. 2022). Transportation simulation software packages (e.g., Eclipse SUMO, MATSim, PTV Vissim, Transims) could potentially support dynamic testing, but most of these tools focus primarily on microscopic traffic simulation (including elements such as traffic lights and pedestrian interactions), a level of detail that is rarely considered for research in our scope. We would like to highlight the work of Maciejewski et al. (2016, 2017), where the authors developed a DVRP extension for MATSim. This extension allows the modeling of a wide variety of DVRPs and the plugging of different algorithms, therefore this tool is indeed suitable for dynamical testing. However, modeling and customization requires familiarity with Java and the relatively complex architecture of MATSim, including a batch of scenario files. Our understanding is that the implementation of the decision making algorithm is also tied to Java.

Based on the above, it is a reasonable goal to develop a standalone simulation tool for DVRPs according to the following criteria. (i) The simulation tool should be based on a generic modeling framework in which the problems can be clearly formulated, thus ensuring the reconstruction of the research. (ii) The framework should be able to model a wide range of DVRPs, that is, the problem aspects and side constraints often occur in the literature should be included by default. (iii) The simulation tool should be easy to use, so it should be much easier to model a problem in it than to implement an entire decision process from scratch. (iv) The simulation tool should be easily customizable and adaptable to individual needs. (v) The implementation of the decision making algorithm should not be tied to a specific programming language, but the simulator should allow communication with it.

We note that, for example, in the field of reinforcement learning – which is also a possible solution approach for DVRPs (see e.g., Hildebrandt et al. 2023) –, the Python package `gym` (or more recently `gymnasium`) has successfully standardized and simplified the testing and comparison of algorithms, which has facilitated the faster introduction and evaluation of new methods (Brockman et al., 2016; Towers et al., 2024).

1.2. Main contributions

Our main goal was to develop a general simulation tool for dynamic vehicle routing. To achieve this, we conducted an extensive literature review and developed a general modeling and simulation framework. Our main contributions are the following.

Literature review on DVRPs. We studied the literature on dynamic vehicle routing to identify those problem aspects and side constraints that are common and should therefore be considered in the development of the framework. For details, see Section 2.

Modeling and simulation framework for DVRPs. We developed a general modeling and simulation framework for dynamic vehicle routing. The framework is suitable for modeling a wide range of DVRPs, primarily pickup-and-delivery problems, but it is easily adaptable to other problems as well. Our *discrete-event based decision process* is a combination of the discrete-event based simulation and the sequential decision process, the latter of which is widely used to formalize DVRPs. For the modeling, we borrowed the route-based representation of Ulmer et al. (2020), but we propose a more detailed model suitable for simulation purposes, see Fig. 1(b). We also standardized and formalized some common aspects of decision making, such as postponing decisions and delaying the departure of vehicles. For details, see Sections 3 and 4.

Open source simulation tool for DVRPs. According to our primary goal, we created an implementation of our simulation framework. The source code of our Python package, called `dvrpsim`, is available online. Dynamic vehicle routing problems can be easily modeled, and the simulator is easily customizable, making it a useful tool for other researchers to dynamically test and evaluate their algorithms for a particular problem. To the best of our knowledge, this is the first simulation tool designed specifically for this purpose. For details, see Section 5.

2. Dynamic vehicle routing

In this section, we provide a brief introduction to dynamic vehicle routing. We also summarize our literature review on dynamic vehicle routing problems. We compiled the reviewed papers in Tables A.1–A.3. The goal of the review was to identify those problem aspects and side constraints that often occur in the literature, therefore, they should be taken into account when developing a general modeling and simulation framework. As the focus is on modeling and simulation, the literature review does not cover problem aspects such as logistic context, objective functions, solution approaches, etc. For such an overview, we refer to the excellent review by Zhang and Van Woensel (2023).

2.1. Dynamic vehicle routing problems

Briefly stated, the well-known *(static) vehicle routing problem* (VRP) aims to determine an optimal set of routes to be performed by a fleet of vehicles to fulfill order requests at different locations within a planning horizon. The problem was introduced more than 60 years ago by Dantzig and Ramser (1959), then generalized by Clarke and Wright (1964), and many variations have appeared since then (e.g., Toth and

Vigo 2002, Eksioglu et al. 2009, Braekers et al. 2016, Zhang et al. 2022).

According to Psaraftis (1980), a vehicle routing problem is characterized as *dynamic*, if the input of the problem is received and updated concurrently with the determination of the routes. The vehicle routes can be redefined in an ongoing fashion. This class of problems is often referred to as *online* or *real-time*. Using the taxonomy of Pillac et al. (2013), a dynamic problem is *stochastic*, if there is some exploitable stochastic knowledge about the dynamically revealed information, and *deterministic* otherwise. Thus, *stochastic dynamic vehicle routing problems* (SDVRPs) are also within the scope of our paper.

In a recent survey, Zhang and Van Woensel (2023) considered three DVRP subcategories by distinguishing three types of order requests. (i) A *pickup and delivery request* consists of a pair of locations, and the serving vehicle must visit the pickup location before going to the delivery location. Table A.1 summarizes the papers we have reviewed on the associated *dynamic pickup-and-delivery problems* (DPDPs). (ii) *Delivery requests* are special pickup and delivery requests because their pickup location refers to a depot. See Table A.2 for our summary on the related *same-day delivery problems* (SDDPs). (iii) A *service request* is associated with only a single location, so the assigned vehicle does not have to visit a specific pickup location (e.g., the depot) before serving the request. See Table A.3 for our overview on *vehicle routing problems with dynamic service requests* (VRPDSRs).

Problems in our scope. In this paper, we focus on the three DVRP subcategories considered by Zhang and Van Woensel (2023). We present our modeling framework primarily for DPDPs (including SDDPs) as we assume that each request has a designated origin and a designated destination, however, with a slight modification the framework is also adaptable to DVRPs with service requests.

Note that Zhang and Van Woensel (2023) identified another DVRP variant in addition to the previous ones, called the *dynamic multi-period VRP* (DMPVRP), which is characterized by multiple planning periods. In this paper, we do not consider these problems. We also do not consider those problems, where the transportation consists of multiple stages, such as *multi-echelon vehicle routing* or *vehicle routing with transshipment*. For a review on these problems, see e.g., Sluijk et al. (2023), Nielsen et al. (2024).

2.2. Sequential decision process

Nowadays, the state-of-the-art approach to modeling DVRPs is the *sequential (or Markov) decision process*. For a thorough introduction, see Ulmer et al. (2020), Soeffker et al. (2022). Briefly stated, at certain time points in the planning horizon, called *decision points*, the decision maker has the opportunity to re-plan the vehicle routes, reflecting on the newly revealed information, see Fig. 1(a). These decision points may be predetermined (e.g., they occur at given intervals), or they can be imposed by certain events (e.g., requesting an order). The sequential decision process steps from decision point to decision point, called *transition*. At a decision point, the decision maker is provided with the current *state*, which describes all the information available to make a decision. The resulted *decision* includes, for example, the updated vehicle routes.

Note in advance that our discrete-event based decision process differs from the sequential decision process in that it explicitly considers events between decision points, see Fig. 1(b). Besides the fact that this approach makes it easier to formalize the dynamic problem in some cases, this level of detail allows us to construct a general, easily customizable simulation framework.

2.3. Problem aspects and side constraints

Now, we present the main aspects and side constraints of dynamic vehicle routing problems that were considered when building our framework. We group these aspects by locations (Section 2.3.1), orders (Section 2.3.2), and vehicles (Section 2.3.3), but there may be some overlap between the groups.

2.3.1. Locations

Location is a collective term for the places that vehicles may visit, such as depots, customers, restaurants, factories, etc., depending on the problem at hand.

Operating network. At this level of logistics planning, vehicles operate on networks. That is, the movement of vehicles is not detailed; they are either at a location (residing at a network node) or on the way (traveling along a network edge). In the latter case, the exact positions of the vehicles are unknown, but their arrival can be calculated from the travel time. In certain cases, vehicle movements are simulated within a real-world road network, such that road crossings also refer to locations (e.g., Ferrucci and Bock 2014, 2015, 2016). Vehicles, especially if they are different types (e.g. drones and trucks), can operate on different networks (e.g., Ulmer and Thomas 2018).

Travel times. Travel times between locations can be arbitrary. For example, travel times can be calculated from the coordinates of the locations (Ulmer et al., 2021), or predefined values (e.g., taken from a map application or based on experience) can be used (Hao et al., 2022). Travel times can be vehicle-dependent, for example, if vehicles have different speeds, and especially if the vehicles operate on different networks (e.g., Ulmer and Thomas 2018). Travel times can also be time-dependent (e.g., Haghani and Jung 2005) or even stochastic (e.g., Schilde et al. 2014).

Docking restrictions. Locations often have limited space for loading or unloading, and sometimes the loading crew creates a bottleneck. Because of these inter-route constraints, vehicles may make each other wait. For example, Hao et al. (2022) proposed a problem, where each factory has a limited number of docking ports, so if a vehicle arrives and there is no port available, the vehicle must wait until a port becomes available.

2.3.2. Orders

Orders are transportation or service requests. The object of transportation can be a variety of products, food (e.g., meal delivery problem), other vehicles (e.g., bike sharing rebalancing problem), or even people (e.g., dial-a-ride problem). Transportation requests typically have an origin (i.e., pickup location) and a destination (i.e., delivery location). In many cases, the terms “order” and “customer” are used interchangeably.

Service times. Various service times may arise when orders are picked up or delivered. Loading and unloading itself may take some time and may even depend on the quantity of orders (e.g., Hao et al. 2022). These times can also be location-dependent (e.g., Ulmer et al. 2019b) or vehicle-dependent (e.g., Ulmer and Thomas 2018). Additional order-independent service times, such as parking or docking, may also occur (e.g., Hao et al. 2022). The above service times can even be stochastic (e.g., Goel et al. 2019), but in many cases they are simply neglected or incorporated in the travel times.

Service time windows. Orders often have *service time windows* for their pickup and/or delivery. Such a time window specifies an *earliest* and a *latest service start time* for the order. Earliest service start times are typically hard constraints, meaning that if a vehicle arrives early at a location, it has to wait until the time window opens, but Schilde et al. (2014), for example, allowed early arrivals. In contrast, latest service start times are often soft constraints, that is, the service can start after the latest required time, however, the tardiness may incur additional costs (e.g., Ulmer et al. 2021). In some rare cases, customers have multiple time windows in the planning horizon (e.g., de Armas and Melián-Batista 2015a,b). Service time windows can be stochastic. For example, in the problem proposed by Srour et al. (2018), customers first preannounce their request with an estimated time window for pickup, which can be changed when the customer confirms the request.

Order cancellation. In some cases, customers can cancel their requests (e.g., Lin et al. 2014, Los et al. 2020). Cancellation is allowed only if the service of the corresponding order has not yet started. After the notification, the decision maker must remove the canceled orders from the vehicle routes. Cancellation is permanent, and canceled orders are no longer dealt with in the given planning horizon.

2.3.3. Vehicles

Vehicle is a collective term for the equipment or people that perform the transportation, such as trucks, drones, drivers, couriers, etc., depending on the problem at hand.

Vehicle fleet. The fleet of vehicles can be either *homogeneous* or *heterogeneous*. In the latter case, vehicles may differ not only in their basic parameters, but also in their operations. For example, Ulmer and Thomas (2018) considered a problem with heterogeneous fleets of drones and trucks that differ not only in their availability, capacity, and travel speed, but also in their requirement for charging and the network on which they operate.

Vehicle capacity. A vehicle is either *capacitated* or *uncapacitated*. In the former case, the total size or quantity of orders loaded on the vehicle must never exceed the capacity of the vehicle. In dial-a-ride or taxi-routing problems, the capacity of the vehicles is the number of non-driver seats, however, in some cases no shared rides are allowed, that is, a vehicle can only carry one passenger (or one passenger group) at a time (e.g., Hyland and Mahmassani 2018). The uncapacitated case is common with those problems where the packages are relatively small and therefore the trunk of the transporting vehicle is not a limiting factor.

Loading rule. Vehicles can be subject to *loading rules*. For example, in Hao et al. (2022), unloading must follow the last-in-first-out (LIFO) rule, i.e., the last loaded order must be unloaded first.

Vehicle availability. Vehicles can also have time windows, representing the working shifts of the drivers (e.g., de Armas and Melián-Batista 2015b, Steever et al. 2019). Sometimes, a time window $[0, L]$ is associated with the depot, also called the *depot deadline*, which gives a latest return time (L) for the vehicles (e.g., Côté et al. 2023).

2.4. Aspects in decision making

Several questions may arise when making decisions. When or how often is it necessary to re-optimize (Section 2.4.1)? Can and should an order be rejected (Section 2.4.2)? Should all decisions be taken as soon as possible, or can certain decisions be postponed (Section 2.4.3)? Should vehicles be sent on their way immediately or is it worth waiting (Section 2.4.4)? Can en route vehicles be diverted or should their destination not be changed (Section 2.4.5)? Can a request be served by multiple routes (Section 2.4.6)?

2.4.1. Decision points

In the case of DVRPs, the decision maker must decide when to process the new dynamic information and update the routes of the vehicles. Most of the articles use three different approaches, namely the decision maker makes a decision either periodically, when a new order request arrives, or when a vehicle arrives at a location, however, there are several other possibilities, and the various approaches can also be combined.

Periodic decision points. In many applications, the planning horizon is divided into predetermined decision epochs, typically of equal length (Δ), i.e., decision points occur periodically. For example, Zolfagharinia and Haughton (2014) re-planned truck routes twice a day ($\Delta = 12$ h). In the framework proposed by Hao et al. (2022) for a dynamic pickup-and-delivery problem, information is updated every 10 min ($\Delta = 10$ min). Bertsimas et al. (2019) re-optimized taxi routes even more frequently ($\Delta = 30$ s).

Decision point on order request. The most common case is that decisions are made when new orders are requested. Ninikas and Minis (2014) also considered a policy where, instead of imposing decision points on every order request, re-optimization would occur after a pre-defined number of requests.

Decision point on vehicle arrival. Often, a decision point is imposed when a vehicle arrives at a location. In some cases, complete order information is not available until arrival, so routes may need to be re-planned prior to the start of service (e.g., Goodson et al. 2016). For some same-day delivery problems, the planned vehicle routes are fixed, so re-optimization occurs only when a vehicle returns to the depot (e.g., Dayarian et al. 2020). In fact, most of the cases decision making is required after the service is finished, but since service times are neglected, it coincides with the arrival. In many approaches, the planned route of a vehicle consists only of the next location to visit, so it is necessary to re-plan the route after the service is finished (e.g., Ulmer et al. 2018, 2019a).

Self-imposed decision points. In some cases, certain decisions can be postponed, which often involves the introduction of self-imposed decision points. That is, if no other event imposes a decision point by a certain time point, then reaching that time will impose one to reconsider the decision. For example, Zhang et al. (2018) considered an orienteering problem in which a traveler must join a waiting queue upon arrival at a location. If the traveler joins the queue, the next decision point is imposed when the size of the queue decreases or a predetermined maximum waiting time elapses, whichever occurs first. Ulmer et al. (2021) investigated a restaurant meal delivery problem, where the assignment of an order to a driver, once made, cannot be altered. Thus, the authors proposed a policy, where the assignment of some non-urgent orders is postponed for a given unit of time, and if no new orders are requested during this period, the expiration of the postponement imposes a decision point. In certain cases, delaying the departure of the vehicles can also cause self-imposed decision points, see later in Section 2.4.4.

2.4.2. Order rejection

In many applications, the decision maker can reject orders, if they are unable or unwilling to fulfill them. The rejection is permanent, and rejected orders are no longer dealt with in the given planning horizon. In practice, rejected orders may be outsourced to a third party or moved to another planning horizon. In the problem proposed by Ehmke and Campbell (2014), the decision maker allows the customer to request an alternative order with a different time window, if the original order is rejected.

2.4.3. Decision postponement

As we touched on in Section 2.4.1, certain decisions can be postponed in some cases. In our interpretation, decision postponement means that certain non-changeable decisions are not made at the current decision point, but are postponed to a later one. For example, if order rejection is allowed, the acceptance/rejection is permanent, therefore some authors do not want to make the decision at the first possible decision point (e.g., Zhang et al. 2018, Voccia et al. 2019). Sometimes, the assignment of orders to vehicles, once made, cannot be altered, so the decision on this assignment is postponed (e.g., Ulmer et al. 2021). Note that the case where the order requests do not impose decision points, and the orders are accepted or rejected at the first decision point after their request, is not considered as decision postponement.

2.4.4. Delaying the departure

In addition to assigning routes to vehicles, it is also important to decide when to send vehicles on their way, since waiting for possible future orders could be beneficial. The two basic waiting strategies, the *drive-first* and the *wait-first*, require a vehicle to departure from its current location at the earliest possible time and at the latest possible time, respectively, but several other waiting strategies have been applied to delay the departure of the vehicles (e.g., Mitrović-Minić and Laporte 2004, Branke et al. 2005, Ichoua et al. 2006).

As mentioned in Section 2.4.1, delaying the departure may involve the use of self-imposed decision points. For example, Voccia et al. (2019) considered a same-day delivery problem, where the depot-to-depot tours cannot be modified during their execution. In their policy, the authors did not start the vehicles immediately after determining their routes, but postponed them for a certain period of time. A decision point was implied at the end of the waiting period, unless another event triggered one in the meantime.

2.4.5. Diversion from the planned route

Due to the dynamic nature of the problem, the decision maker may modify the vehicle routes during execution. Although the majority of papers consider decision making to be instantaneous, in practice it may cover longer periods of time during which the state of the system may change so much (e.g., some vehicles may have already departed) that the decision is no longer feasible with respect to this new state. Therefore, it may be advisable to fix the first parts of the routes, i.e. to make them non-changeable.

In most SDDPs, once the vehicle leaves the depot, its entire route is fixed until it returns to the depot. In some other cases, however, a *preemptive depot return* is allowed, that is, the delivery vehicle can return to the depot before delivering all the orders it is currently carrying (e.g., Ulmer et al. 2019b, Côté et al. 2023).

In general, the next location of a vehicle is fixed. This is especially true when the vehicle is already en route. In some rare cases, however, researchers enable *en route diversion* (e.g., Ulmer et al. 2017, Bosse et al. 2023). In some other cases, vehicle movements are simulated within a real-world road network, where turning on the street is not allowed, so diversions from the current route can only take place at the next road crossing (e.g., Ferrucci and Bock 2014, 2015, 2016). Since in these problems, the road crossings can also be modeled as locations, we do not consider this approach as an en route diversion. In a similar approach, Haferkamp (2024) considered those locations to be deviation points that were located on a traveled shortest path.

2.4.6. Split delivery

Split delivery means that a single request can be served by multiple vehicles (or multiple routes of the same vehicle). Although split delivery is more typical of VRPDSRs (e.g., Schyns 2015, Sarasola et al. 2016), it also occurs in some DPDPs (e.g., Arslan et al. 2021). In the problem formulation of Hao et al. (2022) for a DPDP, orders are inherently split into the smallest deliverable units, and can only be shipped separately if their total demand exceeds the uniform vehicle capacity.

3. A general modeling framework for dynamic vehicle routing I. — Basic concepts

In this section, we propose the basic concept and terminology of our modeling framework. First, we provide an overview of the problems under investigation (Section 3.1). Then, we discuss the main elements in detail, which are the locations (Section 3.2), the orders (Section 3.3), and the vehicles (Section 3.4).

3.1. Main overview: modeling scope

A heterogeneous fleet of vehicles must serve pickup-and-delivery type orders that arrive dynamically in the planning horizon. The pickup/delivery locations can refer to a designated depot, so our modeling framework is suitable for modeling not only DPDPs, but also SDDPs. Various VRPDSRs can be modeled, for example, by specifying coincident pickup and delivery locations. Due to the dynamic nature of the problem, the decision maker has the opportunity to re-plan the vehicle routes at certain decision points. Decision points may be imposed by arbitrary events (e.g., on order request, on vehicle arrival) or may occur periodically. Any parameter of the problem, (e.g., order requests, travel times, etc.) can be deterministic or stochastic.

A service time window can be associated with both the pickup and the delivery of the orders. Both cancellation by the customers and rejection by the decision maker can be handled. In the latter case, the postponement of the decision on acceptance/rejection is also allowed.

Split deliveries are allowed, but in this case, the orders must be split into the smallest deliverable units in advance. It is the decision maker's responsibility to combine and assign them to vehicles according to the splitting rules.

Vehicles can be capacitated or uncapacitated, and may be subject to loading rules. Delaying the departure is possible. The planned routes of the vehicles can be modified during their execution, however, en route diversion is not allowed. Locations may have limited docking capacity, so the vehicles may have to wait for service.

Simulation vs. Decision making. Certain aspects of the problem (Section 2.3) and the decisions (Section 2.4) are not necessarily subject to simulation, but rather to decision making. For example, earliest service start times must obviously be considered by the simulation (since the vehicles must be kept waiting), but latest service start times are the responsibility of the decision maker. Therefore, some aspects, such as order due dates or depot deadlines are not discussed in our modeling framework. However, they can be easily adapted.

3.2. Locations

Locations can refer to different places, such as where orders are to be picked up or delivered, where vehicles are initially located, or they can represent intersections in the real road network. The physical movement of vehicles between locations is not detailed, we just assume that after a vehicle departed for its next location, it will arrive there after a certain amount of time. This travel time must be given or calculable between any two locations that may appear consecutively in the vehicle's route plan, see later in Section 3.4.1. Travel times can be stochastic.

3.3. Orders

Each order o_i has a *pickup location* l_i^p and a *delivery location* l_i^d, which can refer to depots. An order o_i is requested at its *release time* r_i (for static orders, if any, $r_i = 0$). Orders may be associated with an earliest start time for both pickup and delivery. If the vehicle arrives early, it must wait until the latest earliest start time.

3.3.1. Order postponement

In our approach, the decision on an order (i.e., accept/reject) can be postponed until a specific time point. Assume that a decision is made at time t_1 in which an order is postponed until time t_2. The postponement means the following.

Case 1 (postponement is expired). If no decision point is imposed in time interval $[t_1, t_2]$, the postponement of the order is expired. Thus, a decision point will be imposed at t_2, which enables the decision maker to reconsider the order.

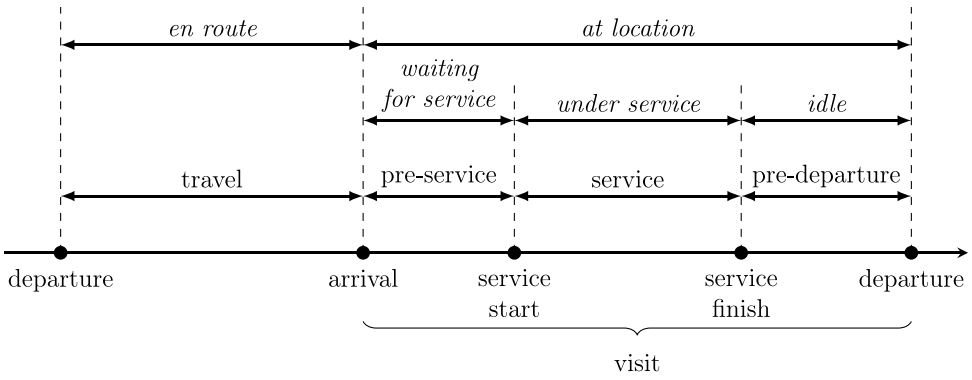


Fig. 2. Vehicle operations between two consecutive departures.

Case 2 (postponement is interrupted). If a decision point is imposed in $[t_1, t_2]$, the postponement of the order will be interrupted at that time. The decision maker may now accept/reject the order, or postpone it again.

3.4. Vehicles

We consider a heterogeneous fleet of vehicles, denoted with $\mathcal{V}$. Each vehicle v is associated with an *initial location* l_v^{init}.

3.4.1. Route plans

The movements of the vehicles are controlled by their route plans. The *route plan* of a vehicle v is a sequence of visits

$$\theta_v = \left(\theta_v^j : j = 1, \dots, \ell_v\right) \text{ with } \theta_v^j = \left(l_v^j, \mathcal{P}_v^j, \mathcal{D}_v^j; est_v^j\right),$$

where each *visit* θ_v^j is specified by a location (l_v^j) to which the vehicle must travel (unless it is currently there), and by (possibly empty) ordered lists ($\mathcal{P}_v^j$ and $\mathcal{D}_v^j$) containing the orders that must be picked up and delivered at the location, respectively. In addition, an *earliest start time* (est_v^j) can be associated with the visit, indicating the earliest time when the vehicle can depart for that location, see later in Section 3.4.3. Route plans will be used later in our decision process to describe the states (Section 4.2) and the decisions (Section 4.4). For an insightful example of route plans we also refer to that section (Section 4.5).

3.4.2. Execution of the route plans

Vehicles – according to their route plan – travel from location to location to perform services there, i.e. to pickup and/or deliver orders. In Fig. 2, we depicted the vehicle operations.

Travel. By *travel*, we mean that the vehicle departs from its *current location* to a specific location, called *destination*. From *departure* to *arrival*, the vehicle is *en route* (i.e., *on the way*). While the vehicle is en route, its exact position is not known. Consequently, the travel cannot be interrupted nor redirected, that is, once the vehicle departed from its current location, it must arrive sooner or later at its destination.

Service. At locations, vehicles perform services. The *service* includes the delivery (unloading) and the pickup (loading) of the corresponding orders, if any, but it may also include other operations, for example, parking or docking. During the service, the vehicle is *under service*. Note that the service may be void, for example, when empty vehicles return back to a depot, or when the location represents a road crossing. Similar to travel, the service cannot be interrupted.

Pre-service. When a vehicle arrives at a location, its service may not start immediately for various reasons. For example, some orders may have an earliest service start time that has not yet passed, some orders may not be ready upon arrival, or some docking restrictions may delay the service. The period between the arrival and the subsequent service start is called *pre-service*. During this period, we say the vehicle is *waiting for service*.

Pre-departure. When the service is finished, the vehicle may not depart immediately for various reasons. For example, the vehicle may have completed its route plan, so the vehicle remains at that location until a new route plan is set. Or the vehicle may have a remaining route, but the start of its execution has been postponed to a later time (see later in Section 3.4.3). The period between the service finish and the subsequent departure is called *pre-departure*. During this period, we say the vehicle is *idle*.

3.4.3. Delaying the departure

Now, we describe our concept for delaying the departure of the vehicles. Assume that vehicle v is ready to departure at time t_1 to its next location, however, an earliest start time t_2 is associated with its next visit. Delaying the departure means the following.

Case 1 (departure postponement is expired). If no decision point is imposed in time interval $[t_1, t_2]$, then the postponement of the vehicle is expired.

Case 1.1 (decision point on departure postponement expiration). If decision points must be imposed on postponement expiration, then a decision point is imposed at t_2, which allows the decision maker, for example, to re-plan the route of the vehicle.

Case 1.2 (no decision point is needed). If no decision points need to be imposed on postponement expiration, then the vehicle departs toward its next location to visit.

Case 2 (departure postponement is interrupted). If a decision point is imposed at $[t_1, t_2]$, then the postponement of the vehicle's departure is interrupted at that time. The decision maker may re-plan the route of the vehicle.

4. A general modeling framework for dynamic vehicle routing II. - Discrete-event based decision process

In this section, we propose our modeling framework, which is called *discrete-event based decision process* reflecting on that it is a combination of the discrete-event simulation and the sequential decision process. The sketch of the process is depicted in Fig. 3. First, we give a main overview of the framework (Section 4.1). Then, we describe the main elements in detail, which are the states (Section 4.2), the events (Section 4.3), and the decisions (Section 4.4).

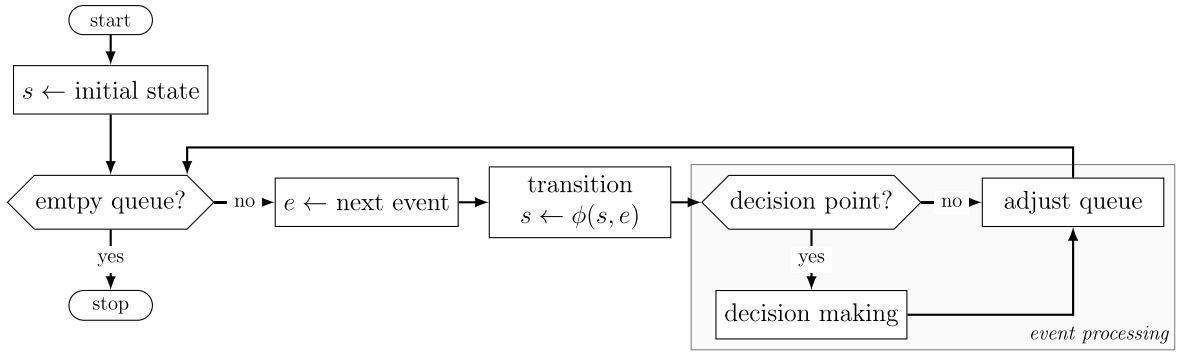


Fig. 3. Sketch of the discrete-event based decision process.

4.1. Main overview

The status of the system – including the current position of vehicles and the current status of orders – is described by *states*. Various *events* (e.g., an order is requested, a vehicle arrives at a location, etc.) occur in the planning horizon. These events are stored in an *event queue*, and the decision process jumps from event to event, always to the one associated with the earliest time. Note that different events can be associated with the same time, and events can be prioritized to establish a processing order between them. There are two special events, the *decision point* event and the *decision enforcement* event. When a decision point event occurs, the decision maker is provided with the current state, and then makes a *decision*. This decision is set when the corresponding decision enforcement event occurs.

4.2. States

A *state* is a tuple

$$s = (t_s, \Phi_s, \Psi_s),$$

where t_s is the current simulation time, $\Phi_s = \{\Phi_{s,v} : v \in \mathcal{V}\}$ is the status of the vehicles, and Ψ_s is the status of the orders, which are discussed in the following. Note that although the system has a state at any given time, since the discrete-event based decision process jumps from event to event, we will later only deal with the states $(s_0, s_1, \ldots)$ induced by these events.

4.2.1. Vehicle status

The status of vehicle v with respect to state s is given as a tuple

$$\Phi_{s,v} = \left(C_{s,v}, \theta_{s,v}\right),$$

where $C_{s,v}$ is the load, i.e., the list of orders currently carried by the vehicle, and

$$\theta_{s,v} = \left(\theta^j_{s,v} : j = 0, \ldots, \ell_{s,v}\right)$$

is the route plan of the vehicle consisting of a sequence of visits, where

$$\theta^0_{s,v} = \left(l^0_{s,v}, \mathcal{P}^0_{s,v}, \mathcal{D}^0_{s,v}; at^0_{s,v}, st^0_{s,v}, ft^0_{s,v}, dt^0_{s,v}\right)$$

is the *origin visit*, and

$$\theta^j_{s,v} = \left(l^j_{s,v}, \mathcal{P}^j_{s,v}, \mathcal{D}^j_{s,v}; est^j_{s,v}\right) \text{ for all } j = 1, \ldots, \ell_{s,v}.$$

are the *next visits*. The origin visit refers to either the current visit of the vehicle, if the vehicle is at a location, or to its previous visit, if the vehicle is en route. Each visit $\theta^j_{s,v}$ consists of a location ($l^j_{s,v}$) and two lists of orders to pickup and to deliver ($\mathcal{P}^j_{s,v}$ and $\mathcal{D}^j_{s,v}$), respectively. The origin visit has four additional elements: the arrival time ($at^0_{s,v}$), the service start time ($st^0_{s,v}$), the service finish time ($ft^0_{s,v}$), and the departure time ($dt^0_{s,v}$) corresponding to the visit. The arrival time is always given, but the other times may not be applicable (denoted by $\varnothing$) if the corresponding event has not happened yet. For example, if the vehicle is currently at a location, then $dt^0_{s,v} = \varnothing$. Otherwise, if $dt^0_{s,v} \neq \varnothing$, the vehicle is currently on the way to its next location $l^1_{s,v}$.

4.2.2. Order status

The status of the orders with respect to state s is given as a tuple

$$\Psi_s = \left(\mathcal{O}^{\text{open}}_s, \mathcal{O}^{\text{canc}}_s\right),$$

where $\mathcal{O}^{\text{open}}_s$ is the set of *open orders* (i.e., already released, neither canceled nor rejected, and not yet delivered orders), and $\mathcal{O}^{\text{canc}}_s$ is the set of those orders that are canceled since the last decision point.

4.2.3. Initial state (s_0)

In the beginning ($t_{s_0} = 0$) vehicles are empty and idle at their initial locations without next visits, i.e., $C_{s_0,v} = \emptyset$ and $\theta_{s_0,v} = ((l^{\text{init}}_v, \emptyset, \emptyset; 0, 0, 0, \varnothing))$ for each vehicle v. No orders are requested yet, that is, $\mathcal{O}^{\text{open}}_{s_0} = \emptyset$ and $\mathcal{O}^{\text{canc}}_{s_0} = \emptyset$.

4.3. Events

Each event is associated with a time. Events are stored in an event queue. When an event occurs, the state of the system changes (Section 4.3.1), and then several other events may be inserted to or removed from the event queue (Section 4.3.2).

Various events can be considered in the model. In the following (we can call it the default model), we consider the following twelve events: *order request*, *order cancellation*, *order pickup*, *order delivery*, *order postponement expiration*, *vehicle arrival*, *vehicle departure*, *service start*, *service finish*, *departure postponement expiration*, *decision point*, and *decision enforcement*.

The first ten events have a medium priority. In contrast, decision point events have a high priority, so if multiple events occur at the same time, decision point events are processed last. In addition, we do not allow multiple decision point events with the same time to be put in the event queue in order to avoid multiple, superfluous decision making. Decision enforcement events have a low priority, so they are processed before all other events.

Uncertainty coming from times (e.g., request time of orders, traveling times, loading times, etc.) can be modeled by adding the corresponding events with "uncertain" (i.e., randomly generated) times to the event queue.

4.3.1. Transition

The decision process steps from event to event, and thus the process transitions from state to state. Formally, *transition* is a function $\phi : \mathcal{S} \times \mathcal{E} \to \mathcal{S}$, where $\mathcal{S}$ is the set of all feasible states, and $\mathcal{E}$ is the set of all events. In fact, only certain events can be considered for a given state (for example, an en route vehicle cannot depart). For the feasibility of states, see Appendix B.

In the following, we formally define the transition from state s_k to the subsequent state $s_{k+1} = \phi(s_k, e)$. Since s_k and s_{k+1} differ only in a few parameters, in order to save space, we only indicate the differences between these states. So first of all, copy the state: $s_{k+1} \leftarrow s_k$. Regardless of the type of e, $t_{s_{k+1}} \leftarrow t$, where t is the time associated with the event.

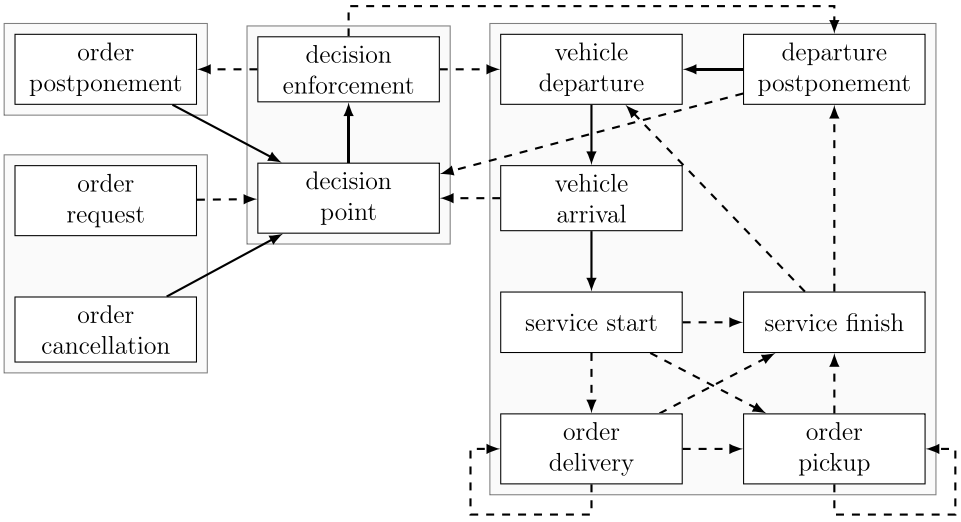


Fig. 4. Events are inductive, meaning that processing one event can cause several new events to be added to or removed from the event queue.

Order request. If event e refers to the request of order o_i, then the order is added to the set of open orders: $\mathcal{O}^{\text{open}}_{s_{k+1}} \leftarrow \mathcal{O}^{\text{open}}_{s_k} \cup \{o_i\}$.

Order pickup. If event e refers to the pickup of order o_i (i.e., the end of loading) by vehicle v, then the order is added to the carrying order list of the vehicle: $C_{s_{k+1},v} \leftarrow C_{s_k,v} \cup \{o_i\}$.

Order delivery. If event e refers to the delivery of order o_i (i.e., the end of unloading) by vehicle v, then the order is removed from the set of open orders, and from the carrying list of the vehicle: $\mathcal{O}^{\text{open}}_{s_{k+1}} \leftarrow \mathcal{O}^{\text{open}}_{s_k} \setminus \{o_i\}$ and $C_{s_{k+1},v} \leftarrow C_{s_k,v} \setminus \{o_i\}$.

Order cancellation. If event e refers to the cancellation of order o_i, then the order is moved from the set of open orders to the list of canceled orders: $\mathcal{O}^{\text{open}}_{s_{k+1}} \leftarrow \mathcal{O}^{\text{open}}_{s_k} \setminus \{o_i\}$ and $\mathcal{O}^{\text{canc}}_{s_{k+1}} \leftarrow \mathcal{O}^{\text{canc}}_{s_k} \cup \{o_i\}$.

Vehicle arrival. If event e refers to the arrival of vehicle v, then the origin visit is removed from the route plan: $\theta^0_{s_{k+1},v} \leftarrow (l^1_{s_k,v}, \mathcal{P}^1_{s_k,v}, \mathcal{D}^1_{s_k,v}; t, \emptyset, \emptyset, \emptyset)$, $\ell_{s_{k+1},v} \leftarrow \ell_{s_k,v} - 1$, and $\theta^j_{s_{k+1},v} \leftarrow \theta^{j+1}_{s_k,v}$ for all $j = 1, \dots, \ell_{s_{k+1},v}$.

Service start. If event e refers to the service start of vehicle v, then the service start time of the origin visit is set: $st^0_{s_{k+1}} \leftarrow t$.

Service finish. If event e refers to the service finish of vehicle v, then the service finish time of the origin visit is set: $ft^0_{s_{k+1}} \leftarrow t$.

Vehicle departure. If event e refers to the departure of vehicle v, then the departure time of the origin visit is set: $dt^0_{s_{k+1}} \leftarrow t$.

Decision enforcement. If event e refers to a decision enforcement, the list of canceled orders is cleared: $\mathcal{O}^{\text{canc}}_{s_{k+1}} \leftarrow \emptyset$, and the decision is enforced (see later in Section 4.4.1).

4.3.2. Event processing

After the transition, the event queue is adjusted, that is, some events may be removed, some new events may be inserted. In Fig. 4, we depict which events can induce which other events. Note that decision points, order request, and order cancellation events can be inserted to the queue from other processes as well.

Decision enforcement. When a decision enforcement event occurs, the associated decision is set (Section 4.4.1). For each postponed order o_i, if any, an order postponement expired event with time $\tilde{pt}_i$ is put into the queue. For each idle vehicle v, if any, a vehicle departure event with the current time (t^{now}) or a departure postponement expiration event associated with the earliest start time ($est^1_{s,v}$) is put into the queue, depending on the next visit of the vehicle.

Decision point. When a decision point event occurs, the decision maker is provided with the current state and returns a decision in response. Then, a decision enforcement event associated with that decision and time t is put into the event queue. Instantaneous decision making can be modeled with $t = t^{\text{now}}$ (where t^{now} is the current time), while real-time time decision making can be modeled with $t' = t^{\text{now}} + \delta$, where δ is the time elapsed during the decision making. In accordance with Sections 3.3.1 and 3.4.3, order postponement expiration and departure postponement expiration events, if any, are removed from the queue.

Order requests and cancellations. When an order request event occurs, a decision point event with time t^{now} may be put into the queue. On the other hand, if an order cancellation event occurs, it may necessary to insert a decision point event into the queue to prevent the canceled order from being picked up.

Vehicle pre-service. After the vehicle arrives at a location, a service start event is put into the event queue. The time associated with the event refers to the time point when the service can be started. Note that this service start time may depend on the service finish of another vehicles.

Vehicle service. A vehicle service may consist of several steps. In the following, we describe the case where orders are first unloaded from the vehicle according to the delivery list, and then orders are loaded to vehicle according to the pickup list. So, after the service starts, order delivery events, then order pickup events, and finally a service finish event are put into the event queue, one after the other, with the previous one inducing the next.

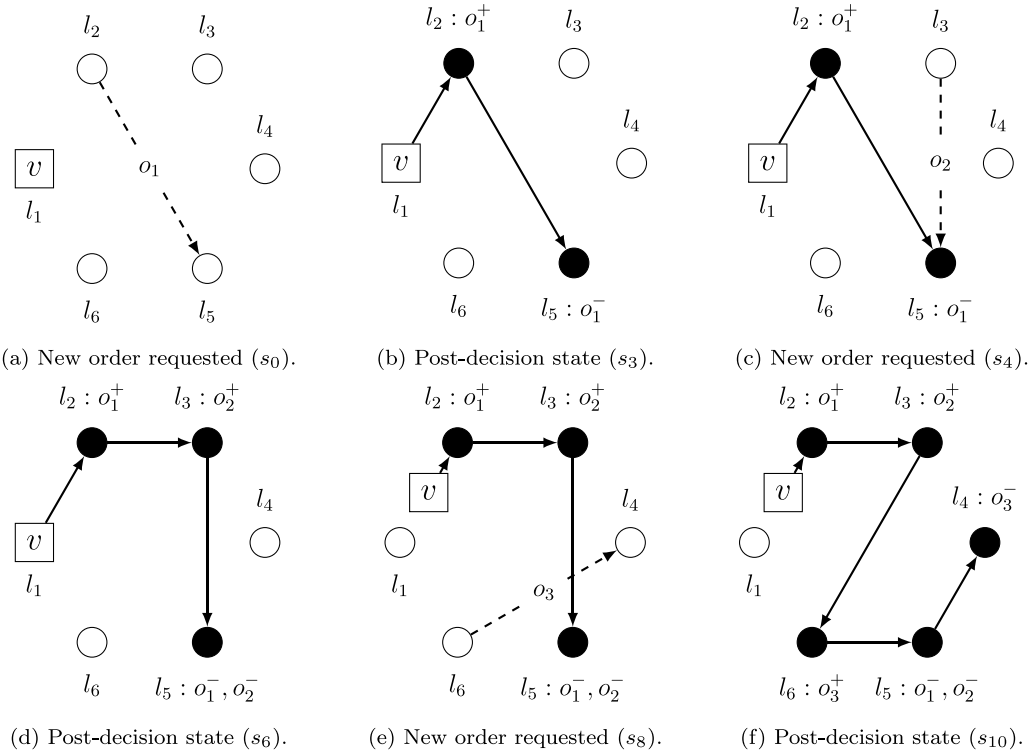
Vehicle pre-departure. After the transition triggered by a service finish event, the vehicle can continue to execute its remaining route plan, if any. (i) If the vehicle has no next visit, there is nothing to do. (ii) If the vehicle has a next visit, and no earliest start time is associated with it, then a departure event is put into the event queue with the actual simulation time (i.e., the vehicle can depart immediately). (iii) If an earliest start time is associated with the vehicle's next visit, than a departure postponement expired event is put into the event queue with that time.

Vehicle travel. After the vehicle departures, a vehicle arrival event – with the time when the vehicle will arrive – is put into the event queue.

4.4. Decisions

A decision is given as a tuple

$$x = \left(\tilde{\Phi}_x, \tilde{\Psi}_x\right),$$



(a) New order requested (s_0). (b) Post-decision state (s_3). (c) New order requested (s_4).

(d) Post-decision state (s_6). (e) New order requested (s_8). (f) Post-decision state (s_{10}).

Fig. 5. Selected states from the following scenario: (s_0) Vehicle v is located at location l_1. (s_1) Order o_1 is requested. (s_2) Decision point is imposed. (s_3) Decision maker creates the route plan. (s_4) Order o_2 is requested. (s_5) Decision point is imposed. (s_6) Decision maker updates the route plan. (s_7) Vehicle is departed. (s_8) Order o_2 is requested. (s_9) Decision point is imposed. (s_{10}) Decision maker updates the route plan.

where $\tilde{\boldsymbol{\Phi}}_x = \{\tilde{\boldsymbol{\Phi}}_{x,v} : v \in \mathcal{V}\}$ is set of the updated route plans, and $\tilde{\Psi}_x$ is the decision on orders, which are discussed in the following.

Decision on orders. The decision on orders is given as a tuple

$$\tilde{\Psi}_x = \left(\tilde{\mathcal{O}}_x^{\text{acc}}, \tilde{\mathcal{O}}_x^{\text{rej}}, \tilde{\mathcal{O}}_x^{\text{post}}\right),$$

where $\tilde{\mathcal{O}}_x^{\text{acc}}$, $\tilde{\mathcal{O}}_x^{\text{rej}}$, and $\tilde{\mathcal{O}}_x^{\text{post}}$ are the set of accepted, rejected, and postponed orders, respectively. Each postponed order $o_i \in \tilde{\mathcal{O}}_x^{\text{post}}$ has a time point $\tilde{pt}_i$ until the decision on the order is postponed (see Section 3.3.1).

Updated route plans. The updated route plan of a vehicle v is a sequence of visits

$$\tilde{\boldsymbol{\Phi}}_{x,v} = \left(\tilde{\theta}_{x,v}^j : j = 0, \dots, \tilde{\ell}_{x,v}\right)$$

with origin visit

$$\tilde{\theta}_{x,v}^0 = \left(\tilde{l}_{x,v}^0, \tilde{\mathcal{P}}_{x,v}^0, \tilde{\mathcal{D}}_{x,v}^0\right)$$

and next visits

$$\tilde{\theta}_{x,v}^j = \left(\tilde{l}_{x,v}^j, \tilde{\mathcal{P}}_{x,v}^j, \tilde{\mathcal{D}}_{x,v}^j; \tilde{est}_{x,v}^j\right) \text{ for all } j = 1, \dots, \tilde{\ell}_{x,v}.$$

Similarly to the states (Section 4.2), each visit $\tilde{\theta}_{x,v}^j$ consists of a location ($\tilde{l}_{x,v}^j$), and pickup and delivery lists ($\tilde{\mathcal{P}}_{x,v}^j$ and $\tilde{\mathcal{D}}_{x,v}^j$). With the exception of the origin visit, each visit can be associated with an earliest start time ($\tilde{est}_{x,v}^j$). The origin visit – more precisely, its pickup and delivery lists – can be modified until the corresponding service starts. If no changes have been made to the previous state, the origin visit may not be given (denoted with $\tilde{\theta}_{x,v}^0 = \varnothing$).

4.4.1. Transition to post-decision state

When a decision x is enforced (see Section 4.3), the decision process transitions to the next state, called *post-decision state* (cf. Powell 2007). Rejected orders, if any, are removed from the list of open orders: $\mathcal{O}_{s_{k+1}}^{\text{open}} \leftarrow \mathcal{O}_{s_k}^{\text{open}} \setminus \tilde{\mathcal{O}}_x^{\text{rej}}$. Then, the route plans of the vehicles are updated. That is, $\theta_{s_{k+1},v}^0 \leftarrow \theta_{s_k,v}^0$ if $\tilde{\theta}_{x,v}^0 = \varnothing$, otherwise $\theta_{s_{k+1},v}^0 \leftarrow (\tilde{\theta}_{x,v}^0; at_{s_k,v}^0, \varnothing, \varnothing, \varnothing)$. Further, $\ell_{s_{k+1},v} \leftarrow \tilde{\ell}_{x,v}$ and $\theta_{s_{k+1},v}^j \leftarrow \tilde{\theta}_{x,v}^j$ for all $j = 1, \dots, \tilde{\ell}_{x,v}$.

4.4.2. Feasibility of decisions

A decision x is *feasible* with respect to state s, if the following constraints are satisfied. For further feasibility conditions, see Appendix B.

Decision on orders. Exactly one decision must be made on each order, that is

$$\tilde{\mathcal{O}}_x^{\text{acc}} \cup \tilde{\mathcal{O}}_x^{\text{rej}} \cup \tilde{\mathcal{O}}_x^{\text{post}} = \mathcal{O}_s^{\text{open}}$$

such that the sets $\tilde{\mathcal{O}}_x^{\text{acc}}$, $\tilde{\mathcal{O}}_x^{\text{rej}}$, and $\tilde{\mathcal{O}}_x^{\text{post}}$ are pairwise disjunctive.

Origin visit. The origin visit of a vehicle v cannot be changed if the service has already started (i.e., the vehicle is either under service or idle or en route).

$$st_{s,v}^0 \neq \varnothing \Rightarrow \tilde{\theta}_{x,v}^0 = \varnothing$$

En route diversion. If vehicle v is en route, its destination cannot be changed.

$$dt_{s,v}^0 \neq \varnothing \Rightarrow \tilde{l}_{x,v}^1 = l_{s,v}^1$$

4.5. Example

In Fig. 5, we depicted selected states from the following scenario for a dynamic pickup-and-delivery problem. (s_0) Vehicle v is initially located at location l_1: $t_{s_0} = 0$, $\mathcal{O}_{s_0}^{\text{open}} = \emptyset$, $\theta_{s_0,v} = ((l_1, (), (); 0, 0, 0, \varnothing))$. ($s_1$) Order o_1 from l_2 to l_5 is requested: $\mathcal{O}_1^{\text{open}} = \{o_1\}$. ($s_2$) A decision point is imposed. The decision maker makes a decision (x_1) that the order is accepted, and the initial route plan is created. However, the departure of the vehicle is delayed until time 10. That is, $\tilde{\mathcal{O}}_{x_1}^{\text{acc}} = \{o_1\}$ and $\tilde{\theta}_{x_1,v} = ((l_2, (o_1), \emptyset; 10), (l_5, \emptyset, (o_1); \varnothing))$. ($s_3$) The decision is enforced:

$\theta_{s_3,v} = ((l_1,\emptyset,\emptyset;0,0,0,\emptyset),\ (l_2,(o_1),();10),\ (l_5,(),(o_1);\emptyset))$. (s_4) Order o_2 from l_3 to l_5 is requested at time 5, thus $t_{s_4} = 5$, $\mathcal{O}^{\text{open}}_{s_4} = \{o_1, o_2\}$. (s_5) A decision point is imposed. The decision maker accepts the order and inserts it into the route plan of the vehicle (x_2). That is, $\tilde{\mathcal{O}}^{\text{acc}}_{x_2} = \{o_1, o_2\}$ and $\tilde{\theta}_{x_2,v} = ((l_2,(o_1),\emptyset;10),\ (l_3,(o_2),\emptyset;\emptyset),\ (l_5,\emptyset,(o_1,o_2);\emptyset))$. (s_6) The decision is enforced: $\theta_{s_6,v} = ((l_1,\emptyset,\emptyset;0,0,0,\emptyset),\ (l_2,(o_1),\emptyset;10),\ (l_3,\emptyset,(o_2);\emptyset),\ (l_5,\emptyset,(o_1,o_2);\emptyset))$. (s_7) The vehicle is departed at time 10, that is, $t_{s_7} = 10$, and $\theta^0_{s_7,v} = (l_1,0,0,0,10)$. (s_8) Order o_3 from l_4 to l_6 is requested at time 12, that is, $t_{s_8} = 12$, and $\mathcal{O}^{\text{open}}_{s_8} = \{o_1, o_2, o_3\}$. (s_9) A decision point is imposed. The decision maker accepts the order and inserts it into the route plan of the vehicle (x_3). That is, $\tilde{\mathcal{O}}^{\text{acc}}_{x_3} = \{o_1, o_2, o_3\}$ and $\tilde{\theta}_{x_3,v} = ((l_2,(o_1),\emptyset;10),\ (l_3,(o_2),\emptyset;\emptyset),\ (l_6,(o_3),\emptyset;\emptyset),\ (l_5,\emptyset,(o_1,o_2);\emptyset),\ (l_4,\emptyset,(o_3);\emptyset))$. (s_{10}) The decision is enforced: $\theta_{s_{10},v} = ((l_1,\emptyset,\emptyset;0,0,0,10),\ (l_2,(o_1),\emptyset;10),\ (l_3,(o_2),\emptyset;\emptyset),\ (l_6,(o_3),\emptyset;\emptyset),\ (l_5,\emptyset,(o_1,o_2);\emptyset),\ (l_4,\emptyset,(o_3);\emptyset))$. (s_{11}) The vehicle is arrived at location l_2 at time 20: $t_{s_{11}} = 20$ and $\theta^0_{s_{11},v} = (l_2,(o_1),\emptyset;20,\emptyset,\emptyset,\emptyset)$. (s_{12}) After the one-minute parking, the service started: $t_{s_{12}} = 21$ and $\theta^0_{s_{12},v} = (l_2,(o_1),\emptyset;20,21,\emptyset,\emptyset)$. (s_{13}) The loading of order o_1 took two minutes: $t_{s_{13}} = 23$, $C_{s_{13},v} = (o_1)$ and $\theta^0_{s_{13},v} = (l_2,(o_1),\emptyset;20,21,23,\emptyset)$. (s_{14}) The vehicle departed immediately to its next location: $\theta^0_{s_{14},v} = (l_2,(o_1),\emptyset;20,21,23,23)$.

5. An open source simulation tool for dynamic vehicle routing

In this section, we briefly present the main components of our simulation framework for dynamic vehicle routing, called `dvrpsim`. Our goal is to provide a concise overview of how to use the simulation package. For an extended, technical description, we refer to the supplementary material. A more detailed tutorial can be found on the webpage of the package: https://sztaki-hu.github.io/dvrpsim/.

5.1. A short introduction

Our simulator is implemented in Python language, however, the implementation of the decision making procedure (also called *external routing algorithm*) is not tied to Python. For the implementation, we used the SimPy package,[1] which is a single-thread process-based discrete-event simulation framework.

5.1.1. Installation

The source code is available at https://github.com/sztaki-hu/dvrpsim. Assuming Python is already installed, the package can also be installed by typing `python -m pip install dvrpsim` at the command prompt.

5.1.2. Modeling (dynamic) vehicle routing problems

To model a vehicle routing problem, the user needs to build a `Model`, and to add the necessary `Locations`, `Orders`, and `Vehicles` that represent the corresponding locations, orders, and vehicles, respectively. These classes have several callback methods, which can be customized to model their desired behavior. The routing callback of the `Model` must be also implemented to connect the external routing algorithm and the simulator.

By starting the simulation (i) each order is requested at its release time; (ii) when a decision point is imposed, the external routing algorithm is called; (iii) once a route plan is set for a vehicle, it begins to execute it. Unless the user implements otherwise, the simulation ends when all orders have been processed (i.e., delivered, canceled, or rejected).

At the end of the simulation, the historical data of the vehicles and orders are available, thus various statistics can be generated. For example, the traveled distance and the total moving/waiting/service/idle time for the vehicles, and the tardiness for the orders are calculated by default.

[1] https://simpy.readthedocs.io/en/latest/

5.1.3. Locations

Each `Location` can optionally be associated with coordinates and a shared resource to model its capacity. The distances and travel times between the locations can be defined and/or used in the corresponding callbacks of the `Vehicles`.

5.1.4. Orders

Each `Order` must be associated with a release time, a pickup location, and a delivery location. There are also several other optional parameters (such as quantity, pickup/delivery time window, pickup/delivery duration, etc.).

During the simulation, each order is requested at its release time, after which the order is available for insertion into a vehicle route. Note that orders can also be created on the fly, while the simulation is running.

An `Order` has several callback methods that are invoked, for example, when the order is requested, rejected, canceled, postponed, picked up, delivered, or when the postponement of the order is expired. By requesting routing in such a callback, the user can model, for example, decision points on order request/cancellation/postponement.

5.1.5. Vehicles

Each `Vehicle` must be associated with an initial location, and there are several other optional parameters (such as capacity, loading rule, etc.). In addition, the travel time callback should be defined that returns the travel time for the vehicle between the corresponding locations.

During the simulation, once a route plan is set for a vehicle as a result of decision making, the vehicle begins to execute it. Recall that the execution procedure of a vehicle consists of four main parts, these are, the pre-departure, the travel, the pre-service, and the service (see Fig. 2). By default, the pre-departure procedure delays the departure of the vehicle when an earliest start time is associated with the next visit. The travel procedure uses the travel time callback to obtain the arrival time at the next location. The pre-service procedure takes into account the earliest service start times of the corresponding orders and the capacity of the corresponding location and, if necessary, makes the vehicle wait accordingly. The service procedure models the unloading and the loading of the corresponding orders.

A `Vehicle` have several callback methods that are invoked, for example, when the vehicle arrives/departs at/from a location, when the service of the vehicle starts/finishes, or when one of its process is interrupted. By requesting routing in such a callback, we can model, for example, decision points on vehicle arrival.

5.1.6. Decision making procedure

The routing callback of the `Model` can be used to connect the external routing algorithm and the simulator. The external routing algorithm can be implemented in arbitrary programming language. Note that the external routing algorithm does not have to be necessary "external", as the algorithm itself can also be implemented in that callback.

At each decision point, a routing callback is invoked, which includes invoking the external routing algorithm. The simulator provides the current state in JSON format, allowing file-based interaction with the external routing algorithm, which is especially useful if the latter is not implemented in Python. The output of the routing algorithm (i.e., the decision) is processed and enforced. Before enforcing the decision, it is possible to check various problem constraints (e.g., the capacity constraints of the vehicles). By default, the simulator assumes instantaneous (i.e., zero time) decision making, but real-time decision making can also be modeled.

5.2. Case studies

As a proof-of-concept, we implemented several examples using our simulator, which are available together with the source code. The following three examples deal with three very different problems with very different problem aspects and constraints, demonstrating that the framework is suitable for modeling a wide range of dynamic vehicle routing problems.

5.2.1. A dynamic pickup-and-delivery problem

A dynamic pickup-and-delivery problem was introduced in a competition organized by the International Conference on Automated Planning and Scheduling in 2021 (ICAPS 2021), see Hao et al. (2022).

Problem overview. There is a fleet of homogeneous vehicles that has to serve pickup-and-delivery order requests which occur over a day. Each order is characterized by a quantity, a pickup factory, a delivery factory, a release time, and a due date. The vehicles can be loaded up to their capacity, while unloading has to follow the last-in-first-out (LIFO) rule. Those, but only those orders whose quantity exceeds the capacity of the vehicles, can be split and delivered separately. The travel times and the distances between the factories are given. Each factory has a given number of docking ports for serving (that is, loading and unloading) the vehicles. Vehicles are served on a first-come-first-served basis. If a vehicle arrives at a factory and all ports are occupied, its service cannot begin immediately, but the vehicle has to join the waiting queue. That is, the vehicle must wait until one of the docking ports becomes free, and no vehicle that arrived earlier is waiting for a port. The objective is to satisfy all the requests such that a combination of tardiness penalties and traveling distances is minimized. Decision points occur in every 10 min.

Proof-of-concept. To model this problem, we used the default `Location` class, where each location is associated with a shared resource to model its docking ports. We also used the default `Order` class, and we split orders into their smallest deliverable units. We inherited a custom `Vehicle` class, where (i) the travel time callback returns the travel times provided in the problem data; (ii) the service procedure is extended to model dock approaching of the vehicles. The latter means that a timeout occurs at the beginning of the service, after which the default service procedure is applied. Capacity and LIFO loading rule are also set for the vehicles. A pre-defined method is used to impose decision points in every 10 min. The form of states and decisions is also modified, so that the `Model` can be connected with the already implemented algorithms for the problem. For more details, we refer to the supplementary material (Section 2.1).

5.2.2. A same-day delivery problem

Voccia et al. (2019) introduced a same-day deliver problem for online purchases. The benchmark instances for their work are publicly available.

Problem overview. The problem is characterized by a fleet of vehicles operating from a depot and by a set of locations. Customers request service throughout the day until a fixed cut-off time. Arrivals of requests are described by a known arrival rate and distribution. Associated with each request is a known service time and a delivery time window at the customer location. Once requests are made, a vehicle at the depot can be assigned requests and leave the depot immediately. Alternatively, a vehicle can wait at the depot before being assigned requests. Once a vehicle leaves the depot, the route for that vehicle is fixed, and the vehicle returns to the depot when it has made all its assigned deliveries. A request is assigned to a third party when it is no longer feasible for the request to be served by a vehicle at the depot or one of the vehicles en route. A decision point is imposed as a result of at least one of the following: (i) a vehicle arrives at the depot; (ii) a vehicle ends its waiting period; (iii) a new request arrives and at least one vehicle is waiting at the depot.

Proof-of-concept. To model this problem, we used the default `Location` and `Order` classes. There is a location for the depot, and there is a separate location for each customer. Each location is associated with latitude and longitude coordinates in order to calculate distances and travel times between locations, when needed. We inherited a custom `Vehicle` class, where the travel time callback returns the travel times calculated on Manhattan-distances. The 'on arrival' and the 'on request' callback of the `Model` are customized to impose decision points on the appropriate events. For more details, we refer to the supplementary material (Section 2.2).

5.2.3. A restaurant meal delivery problem

Ulmer et al. (2021) introduced a restaurant meal delivery problem with random ready times. The benchmark instances for their work are publicly available.

Problem overview. The problem is characterized by a fleet of vehicles that seeks to fulfill a random set of delivery orders that arrive during the finite order horizon from restaurants located in a service area. Orders occur according to a known stochastic process. Each realized order is associated with an order time, a delivery location, a pickup restaurant, and a soft deadline. The time to prepare a customer's food at each restaurant is random. Thus, the driver may need to wait for the order's completion when arriving to a restaurant. The dispatcher determines which orders are assigned to which vehicles. Once made, assignments cannot be altered, therefore, assignments can be postponed. A decision point occurs when a new customer requests service. A decision point can also be self-imposed, which happens when an order is postponed.

Proof-of-concept. To model this problem, we used the default `Location` and `Order` classes. There is a separate location for each restaurant, each customer, and each vehicle. Each location is associated with latitude and longitude coordinates. We inherited a custom `Vehicle` class, where (i) the travel time callback returns the travel times calculated based on Euclidean-distances; (ii) the pre-service procedure is customized to model stochastic ready times. The latter means that when a vehicle (driver) arrives at a restaurant to pick up an order, the callback checks whether it is ready (note that pre-generated ready times are provided in the problem data). If not, the callback schedules an event with the corresponding completion time, and the driver must wait for this event before it can pick up the order. For more details, we refer to the supplementary material (Section 2.3).

6. Conclusion

In this paper, we focused on developing a simulation tool designed to model a wide range of dynamic vehicle routing problems (DVRPs) to support the dynamic testing of different solution methods.

We began by conducting an extensive literature review to identify the key aspects and common constraints in DVRPs that should be considered in the modeling framework. Based on these findings, we developed a general modeling and simulation framework tailored for simulation purposes. Finally, we have created an implementation of the framework and made it freely available. As a proof-of-concept, we have implemented several examples with our framework. These case studies deal with different problems with very different problem aspects and constraints, demonstrating that the framework is suitable for modeling a wide range of dynamic vehicle routing problems.

Our plan for the future is to maintain and improve the framework. We will try to answer all user questions and be open to ideas for improvement (e.g., new features). We would like to create more thorough documentation on the package website and to add more case studies to the example collection. We have several ideas for further improvements, the first of which is to implement more extensive checking of possible route feasibility constraints, and to provide more detailed statistics. We also want to use this framework in our research on various dynamic vehicle routing problems.

Table A.1
Problem and decision making aspects for DPDPs.

Paper	VEH	CAP	TW	CAN	DPs	DEL	REJ	PP	ERD
Ferrucci and Bock (2014)	He	Yes	S	–	P	–	Yes	–	(Yes)
Schilde et al. (2014)	Ho	Yes	S	–	P	Yes	–	–	–
Zolfagharinia and Haughton (2014)	He	Yes	H	–	P	–	Yes	–	(Yes)
Ma et al. (2015)	He	Yes	H	–	OR	–	Yes	–	–
Muñoz-Carpintero et al. (2015)	He	Yes	–	–	OR	–	–	–	–
Sayarshad and Chow (2015)	He	Yes	–	Yes	OR	–	–	–	–
Wang and Kopfer (2015)	He	Yes	H	–	OR/P	–	Yes	–	–
Vonolfen and Affenzeller (2016)	Ho	Yes	H	–	OR	Yes	–	–	–
Zolfagharinia and Haughton (2016)	He	Yes	H	–	P	Yes	Yes	–	–
Tirado and Hvattum (2017a)	He	Yes	H	–	OR, VA	Yes	Yes	Yes	–
Tirado and Hvattum (2017b)	He	Yes	H	–	OR, VA	–	Yes	Yes	–
Hyland and Mahmassani (2018)	Ho	Yes	–	–	P	–	–	–	–
Sayarshad and Oliver Gao (2018)	He	Yes	–	–	OR	–	–	–	–
Srour et al. (2018)	Ho	Yes	H	–	OM	Yes	Yes	–	–
Arslan et al. (2019)	He	Yes	H	–	NI	–	–	–	–
Bertsimas et al. (2019)	Ho	Yes	H	–	P	–	Yes	Yes	–
Györgyi and Kis (2019)	Ho	Yes	H	–	OM	Yes	Yes	–	–
He et al. (2019)	Ho	Yes	S	–	OR	–	–	–	–
Liu (2019)	He	Yes	–	–	P	–	–	Yes	–
Steever et al. (2019)	He	Yes	S	–	OR	–	–	–	–
Duan et al. (2020)	Ho	Yes	H	–	P	–	Yes	–	–
Karami et al. (2020)	Ho	–	S	–	P	–	–	–	–
Los et al. (2020)	He	Yes	H	Yes	OR	Yes	Yes	–	–
Arslan et al. (2021)	Ho	Yes	H	–	OR	–	Yes	–	–
Tafreshian et al. (2021)	Ho	Yes	H	–	P	Yes	Yes	–	–
Ulmer et al. (2021)	Ho	–	S	–	OR, SI	–	–	Yes	–
Ghiani et al. (2022)	Ho	–	–	–	OR	–	–	–	–
Haferkamp and Ehmke (2022)	Ho	–	H	–	OR	–	Yes	–	–
Kullman et al. (2022)	Ho	–	H	–	OR, VA	–	Yes	–	–
Hao et al. (2022)	Ho	Yes	S	–	P	–	–	–	–
Ackermann and Rieck (2023)	Ho	Yes	–	–	OR, VA, SI	–	Yes	Yes	–
Auad et al. (2023)	Ho	Yes	S	–	P	–	–	Yes	–
Bosse et al. (2023)	Ho	Yes	–	–	OR	–	Yes	–	Yes
Dieter et al. (2023)	Ho	Yes	–	–	OR	–	–	–	–
Heitmann et al. (2023)	Ho	Yes	H	–	OR	–	Yes	–	–
Ackva and Ulmer (2024)	Ho	Yes	H	–	OR	Yes	Yes	–	–
Jeong and Moon (2024)	Ho	Yes	–	–	OR	–	–	–	–
Heitmann et al. (2024)	Ho	Yes	H	–	OR	–	Yes	–	–
Haferkamp (2024)	Ho	Yes	H	(Yes)	OR	–	–	–	(Yes)

Table A.2
Problem and decision making aspects for SDDPs.

Paper	VEH	CAP	TW	CAN	DPs	DEL	REJ	PP	ERD
Ehmke and Campbell (2014)	Ho	–	H	–	OR	–	Yes	–	–
Klapp et al. (2018a)	1	–	–	–	P	–	Yes	Yes	–
Klapp et al. (2018b)	1	–	–	–	P	Yes	Yes	–	–
Ulmer and Thomas (2018)	He	Yes	H	–	OR	–	Yes	–	–
Ulmer and Streng (2019)	Ho	Yes	–	–	P	–	–	Yes	–
Ulmer et al. (2019b)	1	–	–	–	VA	–	Yes	–	–
van Heeswijk et al. (2019)	Ho	Yes	H	–	P	–	–	Yes	–
Voccia et al. (2019)	Ho	–	H	–	OR, VA, SI	Yes	Yes	Yes	–
Dayarian and Savelsbergh (2020)	He	Yes	S	–	P, VA	Yes	–	Yes	–
Dayarian et al. (2020)	He	Yes	H	–	VA	Yes	Yes	–	–
Klapp et al. (2020)	Ho	–	–	–	P, OR	Yes	Yes	–	–
Ulmer (2020)	Ho	–	H	–	OR	–	Yes	–	–
Chen et al. (2022)	He	Yes	H	–	OR	–	Yes	–	–
Chen et al. (2023)	He	–	H	–	OR	–	Yes	–	–
Côté et al. (2023)	Ho	–	H	–	OR, VA, SI	Yes	Yes	–	–
Liu and Luo (2023)	Ho	Yes	H	–	P	–	–	(Yes)	–

Table A.3
Problem and decision making aspects for VRPDSRs.

Paper	VEH	CAP	TW	CAN	DPs	DEL	REJ	PP	ERD
Lin et al. (2014)	Ho	Yes	H	Yes	OR	–	–	–	–
Ninikas and Minis (2014)	Ho	Yes	H	–	OR	–	–	–	–
Ferrucci and Bock (2015)	Ho	–	S	–	P	–	–	–	(Yes)
de Armas and Melián-Batista (2015b)	He	Yes	S	–	OR	–	Yes	–	–
Schyns (2015)	He	Yes	H	Yes	NI	–	–	–	–
Ferrucci and Bock (2016)	Ho	–	S	–	P	Yes	–	–	(Yes)
Sarasola et al. (2016)	Ho	Yes	–	–	P	–	–	–	–
Angelelli et al. (2016)	1	–	–	–	VA	–	–	–	–
Goodson et al. (2016)	Ho	Yes	–	–	VA	–	–	–	–
Ng et al. (2017)	Ho	Yes	–	–	VA	–	–	–	–
Ulmer et al. (2017)	1	–	–	–	OR, VA, SI	–	Yes	Yes	Yes
Pillac et al. (2018)	He	–	H	–	OR,VA	Yes	Yes	–	–
Ulmer et al. (2018)	1	–	–	–	VA	Yes	Yes	–	–
Zhang et al. (2018)	1	–	H	–	VA, SI	–	Yes	Yes	–
Ulmer (2019)	1	–	–	–	VA	–	Yes	–	–
Ulmer et al. (2019a)	1	–	–	–	VA	(Yes)	Yes	–	–
Bono et al. (2021)	Ho	Yes	S	–	VA	–	–	–	–
Xiang et al. (2022)	Ho	Yes	–	–	P	–	–	–	–
Zhang et al. (2023)	Ho	–	–	–	OR	–	Yes	–	–
Soeffker et al. (2024)	Ho	–	–	–	OR	–	Yes	–	–

CRediT authorship contribution statement

Markó Horváth: Writing – original draft, Visualization, Validation, Supervision, Software, Methodology, Investigation, Conceptualization. **Tímea Tamási:** Writing – original draft, Validation, Software, Methodology, Investigation, Conceptualization.

Declaration of Generative AI and AI-assisted technologies in the writing process

During the preparation of this work the authors used deepL in order to check the accuracy of the English text they have created. After using this tool, the authors reviewed and edited the content as needed and take full responsibility for the content of the publication.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research has been supported by the TKP2021-NKTA-01 NRDIO grant on “Research on cooperative production and logistics systems to support a competitive and sustainable economy”. Markó Horváth acknowledges the support of the János Bolyai Research Scholarship.

Appendix A. Tables for literature review

In Tables A.1–A.3 we compiled the reviewed papers. Abbreviations stand for the following. Vehicles (VEH): Single (1), Homogeneous fleet (Ho), Heterogeneous fleet (He). Capacitated vehicles (CAP). Order time-windows (TW): Soft (S), Hard (H). Order cancellation (CAN). Decision points (DPs): Periodic (P), Order request (OR), Vehicle arrival (VA), Self-imposed (SI), New information (NI), Order modification (OM). Delaying the departure (DEL). Order rejection (REJ). Decision postponement (PP). En route diversion (ERD).

Appendix B. Feasibility of states and decisions

A state s is *feasible* if the following constraints are satisfied. A decision x is *feasible with respect to* the feasible state s if, in addition to the constraints described in Section 4.4.2, the post-decision state $\phi(s,x)$ is feasible.

B.1. General constraints

Regardless of the problem, the following constraints must always be taken into account.

Assigned orders. Only open orders can be assigned to vehicles.

$$\bigcup_{v\in\mathcal{V}}\left(C_{s,v}\cup\bigcup_{j=0}^{\ell_{s,v}}\left(\mathcal{P}^j_{s,v}\cup\mathcal{D}^j_{s,v}\right)\right)\subseteq\mathcal{O}_s^{\text{open}}$$

Pickup and delivery locations. Orders can only be picked up at their pickup location ($l^p_{\cdot}$), and can only be delivered at their delivery location ($l^d_{\cdot}$).

$$o_i\in\mathcal{P}^j_{s,v}\Rightarrow l^j_{s,v}=l^p_i$$

$$o_i\in\mathcal{D}^j_{s,v}\Rightarrow l^j_{s,v}=l^d_i$$

Pickup and delivery with the same vehicle. Orders must be delivered by the same vehicle that picked them up.

$$o_i\in\mathcal{D}^j_{s,v}\Rightarrow o_i\in C_{s,v}\cup\bigcup_{k=0}^{j-1}\mathcal{P}^k_{s,v}$$

Pickup and deliver only once. Orders can only be picked up and delivered once. That is, the sets $C_{s,v}$ and $\mathcal{P}^j_{s,v}$ $(j=0,\dots,\ell_{s,v})$ must be pairwise disjunctive for each vehicle v. Similarly, for each vehicle v, the sets $\mathcal{D}^j_{s,v}$ $(j=0,\dots,\ell_{s,v})$ must be pairwise disjunctive.

B.2. Problem specific constraints

There may be several other constraints for a particular problem at hand (e.g., capacity constraints, loading rules).

Capacity constraints. If vehicle v is capacitated, then the total quantity of the loaded orders cannot exceed its capacity Q_v That is,

$$\sum_{o_i\in C_{s,v}}q_i+\sum_{j=0}^{j'}\left(\sum_{o_i\in\mathcal{D}^j_{s,v}}q_i-\sum_{o_i\in\mathcal{P}^j_{s,v}}q_i\right)\le Q_v\quad\text{for all } j'=0,\dots,\ell_{s,v},$$

where it is assumed that unloading takes place first and then loading takes place afterwards.

Appendix C. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.ejtl.2025.100159.

References

Ackermann, C., Rieck, J., 2023. A novel repositioning approach and analysis for dynamic ride-hailing problems. EURO J. Transp. Logist. 12, 100109.

Ackva, C., Ulmer, M.W., 2024. Consistent routing for local same-day delivery via micro-hubs. OR Spectrum 46 (2), 375–409.

Angelelli, E., Mansini, R., Vindigni, M., 2016. The stochastic and dynamic traveling purchaser problem. Transp. Sci. (ISSN: 0041-1655) 50 (2), 642–658. http://dx.doi.org/10.1287/trsc.2015.0627, Publisher: INFORMS.

Arslan, A.M., Agatz, N., Klapp, M.A., 2021. Operational strategies for on-demand personal shopper services. Transp. Res. Part C: Emerg. Technol. 130, 103320.

Arslan, A.M., Agatz, N., Kroon, L., Zuidwijk, R., 2019. Crowdsourced delivery—A dynamic pickup and delivery problem with ad hoc drivers. Transp. Sci. (ISSN: 0041-1655) 53 (1), 222–235. http://dx.doi.org/10.1287/trsc.2017.0803.

Auad, R., Erera, A., Savelsbergh, M., 2023. Courier satisfaction in rapid delivery systems using dynamic operating regions. Omega 121, 102917.

Bektaş, T., Repoussis, P.P., Tarantilis, C.D., 2014. Chapter 11: dynamic vehicle routing problems. In: Vehicle Routing: Problems, Methods, and Applications, Second Edition. SIAM, pp. 299–347.

Berbeglia, G., Cordeau, J.-F., Laporte, G., 2010. Dynamic pickup and delivery problems. European J. Oper. Res. 202 (1), 8–15.

Bertsimas, D., Jaillet, P., Martin, S., 2019. Online vehicle routing: The edge of optimization in large-scale applications. Oper. Res. (ISSN: 0030-364X) 67 (1), 143–162. http://dx.doi.org/10.1287/opre.2018.1763, Publisher: INFORMS.

Bono, G., Dibangoye, J.S., Simonin, O., Matignon, L., Pereyron, F., 2021. Solving multi-agent routing problems using deep attention mechanisms. IEEE Trans. Intell. Transp. Syst. (ISSN: 1558-0016) 22 (12), 7804–7813. http://dx.doi.org/10.1109/TITS.2020.3009289, Conference Name: IEEE Transactions on Intelligent Transportation Systems.

Bosse, A., Ulmer, M.W., Manni, E., Mattfeld, D.C., 2023. Dynamic priority rules for combining on-demand passenger transportation and transportation of goods. European J. Oper. Res. 309 (1), 399–408.

Braekers, K., Ramaekers, K., Van Nieuwenhuyse, I., 2016. The vehicle routing problem: State of the art classification and review. Comput. Ind. Eng. 99, 300–313.

Branke, J., Middendorf, M., Noeth, G., Dessouky, M., 2005. Waiting strategies for dynamic vehicle routing. Transp. Sci. 39 (3), 298–312.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., Zaremba, W., 2016. Openai gym. arXiv preprint arXiv:1606.01540.

Chen, X., Ulmer, M.W., Thomas, B.W., 2022. Deep Q-learning for same-day delivery with vehicles and drones. European J. Oper. Res. (ISSN: 0377-2217) 298 (3), 939–952. http://dx.doi.org/10.1016/j.ejor.2021.06.021.

Chen, X., Wang, T., Thomas, B.W., Ulmer, M.W., 2023. Same-day delivery with fair customer service. European J. Oper. Res. 308 (2), 738–751.

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

Côté, J.-F., de Queiroz, T.A., Gallesi, F., Iori, M., 2023. A branch-and-regret algorithm for the same-day delivery problem. Transp. Res. Part E: Logist. Transp. Rev. 177, 103226.

Dantzig, G.B., Ramser, J.H., 1959. The truck dispatching problem. Manag. Sci. 6 (1), 80–91.

Dayarian, I., Savelsbergh, M., 2020. Crowdshipping and same-day delivery: Employing in-store customers to deliver online orders. Prod. Oper. Manage. (ISSN: 1937-5956) 29 (9), 2153–2174. http://dx.doi.org/10.1111/poms.13219.

Dayarian, I., Savelsbergh, M., Clarke, J.-P., 2020. Same-day delivery with drone resupply. Transp. Sci. (ISSN: 0041-1655) 54 (1), 229–249. http://dx.doi.org/10.1287/trsc.2019.0944, Publisher: INFORMS.

de Armas, J., Melián-Batista, B., 2015a. Constrained dynamic vehicle routing problems with time windows. Soft Comput. 19, 2481–2498.

de Armas, J., Melián-Batista, B., 2015b. Variable neighborhood search for a dynamic rich vehicle routing problem with time windows. Comput. Ind. Eng. (ISSN: 0360-8352) 85, 120–131. http://dx.doi.org/10.1016/j.cie.2015.03.006.

Dieter, P., Stumpe, M., Ulmer, M.W., Schryen, G., 2023. Anticipatory assignment of passengers to meeting points for taxi-ridesharing. Transp. Res. Part D: Transp. Environ. 121, 103832.

Duan, L., Wei, Y., Zhang, J., Xia, Y., 2020. Centralized and decentralized autonomous dispatching strategy for dynamic autonomous taxi operation in hybrid request mode. Transp. Res. Part C: Emerg. Technol. (ISSN: 0968-090X) 111, 397–420. http://dx.doi.org/10.1016/j.trc.2019.12.020.

Ehmke, J.F., Campbell, A.M., 2014. Customer acceptance mechanisms for home deliveries in metropolitan areas. European J. Oper. Res. (ISSN: 0377-2217) 233 (1), 193–207. http://dx.doi.org/10.1016/j.ejor.2013.08.028.

Eksioglu, B., Vural, A.V., Reisman, A., 2009. The vehicle routing problem: A taxonomic review. Comput. Ind. Eng. 57 (4), 1472–1483.

Ferrucci, F., Bock, S., 2014. Real-time control of express pickup and delivery processes in a dynamic environment. Transp. Res. Part B: Methodol. (ISSN: 0191-2615) 63, 1–14. http://dx.doi.org/10.1016/j.trb.2014.02.001.

Ferrucci, F., Bock, S., 2015. A general approach for controlling vehicle en-route diversions in dynamic vehicle routing problems. Transp. Res. Part B: Methodol. (ISSN: 0191-2615) 77, 76–87. http://dx.doi.org/10.1016/j.trb.2015.03.003.

Ferrucci, F., Bock, S., 2016. Pro-active real-time routing in applications with multiple request patterns. European J. Oper. Res. (ISSN: 0377-2217) 253 (2), 356–371. http://dx.doi.org/10.1016/j.ejor.2016.02.016.

Ghiani, G., Manni, A., Manni, E., 2022. A scalable anticipatory policy for the dynamic pickup and delivery problem. Comput. Oper. Res. (ISSN: 0305-0548) 147, 105943. http://dx.doi.org/10.1016/j.cor.2022.105943.

Goel, R., Maini, R., Bansal, S., 2019. Vehicle routing problem with time windows having stochastic customers demands and stochastic service times: Modelling and solution. J. Comput. Sci. 34, 1–10.

Goodson, J.C., Thomas, B.W., Ohlmann, J.W., 2016. Restocking-based rollout policies for the vehicle routing problem with stochastic demand and duration limits. Transp. Sci. (ISSN: 0041-1655) 50 (2), 591–607. http://dx.doi.org/10.1287/trsc.2015.0591, Publisher: INFORMS.

Györgyi, P., Kis, T., 2019. A probabilistic approach to pickup and delivery problems with time window uncertainty. European J. Oper. Res. 274 (3), 909–923.

Haferkamp, J., 2024. Design of multi-optional pickup time offers in ride-sharing systems. EURO J. Transp. Logist. 100134.

Haferkamp, J., Ehmke, J.F., 2022. Effectiveness of demand and fulfillment control in dynamic fleet management of ride-sharing systems. Networks (ISSN: 1097-0037) 79 (3), 314–337. http://dx.doi.org/10.1002/net.22062.

Haghani, A., Jung, S., 2005. A dynamic vehicle routing problem with time-dependent travel times. Comput. Oper. Res. 32 (11), 2959–2986.

Hao, J., Lu, J., Li, X., Tong, X., Xiang, X., Yuan, M., Zhuo, H.H., 2022. Introduction to the dynamic pickup and delivery problem benchmark – ICAPS 2021 competition.

He, Z., Han, G., Cheng, T.C.E., Fan, B., Dong, J., 2019. Evolutionary food quality and location strategies for restaurants in competitive online-to-offline food ordering and delivery markets: An agent-based approach. Int. J. Prod. Econ. (ISSN: 0925-5273) 215, 61–72. http://dx.doi.org/10.1016/j.ijpe.2018.05.008.

Heitmann, R.-J.O., Soeffker, N., Klawonn, F., Ulmer, M.W., Mattfeld, D.C., 2024. Accelerating value function approximations for dynamic dial-a-ride problems via dimensionality reductions. Comput. Oper. Res. 167, 106639.

Heitmann, R.-J.O., Soeffker, N., Ulmer, M.W., Mattfeld, D.C., 2023. Combining value function approximation and multiple scenario approach for the effective management of ride-hailing services. EURO J. Transp. Logist. 12, 100104.

Hildebrandt, F.D., Thomas, B.W., Ulmer, M.W., 2023. Opportunities for reinforcement learning in stochastic dynamic vehicle routing. Comput. Oper. Res. 150, 106071.

Hyland, M., Mahmassani, H.S., 2018. Dynamic autonomous vehicle fleet operations: Optimization-based strategies to assign AVs to immediate traveler demand requests. Transp. Res. Part C: Emerg. Technol. (ISSN: 0968-090X) 92, 278–297. http://dx.doi.org/10.1016/j.trc.2018.05.003.

Ichoua, S., Gendreau, M., Potvin, J.-Y., 2006. Exploiting knowledge about future demands for real-time vehicle dispatching. Transp. Sci. 40 (2), 211–225.

Jeong, J., Moon, I., 2024. Dynamic pickup and delivery problem for autonomous delivery robots in an airport terminal. Comput. Ind. Eng. 196, 110476.

Karami, F., Vancroonenburg, W., Vanden Berghe, G., 2020. A periodic optimization approach to dynamic pickup and delivery problems with time windows. J. Sched. (ISSN: 1094-6136) 23 (6), 711–731. http://dx.doi.org/10.1007/s10951-020-00650-x.

Klapp, M.A., Erera, A.L., Toriello, A., 2018a. The dynamic dispatch waves problem for same-day delivery. European J. Oper. Res. (ISSN: 0377-2217) 271 (2), 519–534. http://dx.doi.org/10.1016/j.ejor.2018.05.032.

Klapp, M.A., Erera, A.L., Toriello, A., 2018b. The one-dimensional dynamic dispatch waves problem. Transp. Sci. (ISSN: 0041-1655) 52 (2), 402–415. http://dx.doi.org/10.1287/trsc.2016.0682, Publisher: INFORMS.

Klapp, M.A., Erera, A.L., Toriello, A., 2020. Request acceptance in same-day delivery. Transp. Res. Part E: Logist. Transp. Rev. (ISSN: 1366-5545) 143, 102083. http://dx.doi.org/10.1016/j.tre.2020.102083.

Kullman, N.D., Cousineau, M., Goodson, J.C., Mendoza, J.E., 2022. Dynamic ride-hailing with electric vehicles. Transp. Sci. 56 (3), 775–794.

Lin, C., Choy, K.L., Ho, G.T.S., Lam, H.Y., Pang, G.K.H., Chin, K.S., 2014. A decision support system for optimizing dynamic courier routing operations. Expert Syst. Appl. (ISSN: 0957-4174) 41 (15), 6917–6933. http://dx.doi.org/10.1016/j.eswa.2014.04.036.

Liu, Y., 2019. An optimization-driven dynamic vehicle routing algorithm for on-demand meal delivery using drones. Comput. Oper. Res. (ISSN: 03050548) 111, 1–20. http://dx.doi.org/10.1016/j.cor.2019.05.024.

Liu, S., Luo, Z., 2023. On-demand delivery from stores: Dynamic dispatching and routing with random demand. Manuf. Serv. Oper. Manag. 25 (2), 595–612.

Los, J., Schulte, F., Spaan, M.T.J., Negenborn, R.R., 2020. The value of information sharing for platform-based collaborative vehicle routing. Transp. Res. Part E: Logist. Transp. Rev. (ISSN: 1366-5545) 141, 102011. http://dx.doi.org/10.1016/j.tre.2020.102011.

Ma, S., Zheng, Y., Wolfson, O., 2015. Real-time city-scale taxi ridesharing. IEEE Trans. Knowl. Data Eng. (ISSN: 1558-2191) 27 (7), 1782–1795. http://dx.doi.org/10.1109/TKDE.2014.2334313, Conference Name: IEEE Transactions on Knowledge and Data Engineering.

Maciejewski, M., Bischoff, J., Hörl, S., Nagel, K., 2017. Towards a testbed for dynamic vehicle routing algorithms. In: Highlights of Practical Applications of Cyber-Physical Multi-Agent Systems: International Workshops of PAAMS 2017, Porto, Portugal, June 21–23, 2017, Proceedings 15. Springer, pp. 69–79.

Maciejewski, M., Horni, A., Nagel, K., Axhausen, K.W., 2016. Dynamic transport services. In: The multi-agent Transport Simulation MATSim, vol. 23, Ubiquity Press London, pp. 145–152.

Mardešić, N., Erdelić, T., Carić, T., Durasević, M., 2023. Review of stochastic dynamic vehicle routing in the evolving urban logistics environment. Mathematics 12 (1), 28.

Mitrović-Minić, S., Laporte, G., 2004. Waiting strategies for the dynamic pickup and delivery problem with time windows. Transp. Res. Part B: Methodol. 38 (7), 635–655.

Muñoz-Carpintero, D., Sáez, D., Cortés, C.E., Núñez, A., 2015. A methodology based on evolutionary algorithms to solve a dynamic pickup and delivery problem under a hybrid predictive control approach. Transp. Sci. (ISSN: 0041-1655) 49 (2), 239–253. http://dx.doi.org/10.1287/trsc.2014.0569, Publisher: INFORMS.

Ng, K.K.H., Lee, C.K.M., Zhang, S.Z., Wu, K., Ho, W., 2017. A multiple colonies artificial bee colony algorithm for a capacitated vehicle routing problem and re-routing strategies under time-dependent traffic congestion. Comput. Ind. Eng. (ISSN: 0360-8352) 109, 151–168. http://dx.doi.org/10.1016/j.cie.2017.05.004.

Nielsen, P., Dahanayaka, M., Perera, H.N., Thibbotuwawa, A., Kilic, D.K., 2024. A systematic review of vehicle routing problems and models in multi-echelon distribution networks. Supply Chain Anal. 100072.

Ninikas, G., Minis, I., 2014. Reoptimization strategies for a dynamic vehicle routing problem with mixed backhauls. Networks (ISSN: 1097-0037) 64 (3), 214–231. http://dx.doi.org/10.1002/net.21567.

Pillac, V., Gendreau, M., Guéret, C., Medaglia, A.L., 2013. A review of dynamic vehicle routing problems. European J. Oper. Res. 225 (1), 1–11.

Pillac, V., Guéret, C., Medaglia, A.L., 2018. A fast reoptimization approach for the dynamic technician routing and scheduling problem. In: Amodeo, L., Talbi, E.-G., Yalaoui, F. (Eds.), Recent Developments in Metaheuristics. Springer International Publishing, ISBN: 978-3-319-58253-5, pp. 347–367. http://dx.doi.org/10.1007/978-3-319-58253-5_20.

Powell, W.B., 2007. Approximate Dynamic Programming: Solving the Curses of Dimensionality, vol. 703, John Wiley & Sons.

Psaraftis, H.N., 1980. A dynamic programming solution to the single vehicle many-to-many immediate request dial-a-ride problem. Transp. Sci. 14 (2), 130–154.

Psaraftis, H.N., Wen, M., Kontovas, C.A., 2016. Dynamic vehicle routing problems: Three decades and counting. Networks 67 (1), 3–31.

Rios, B.H.O., Xavier, E.C., Miyazawa, F.K., Amorim, P., Curcio, E., Santos, M.J., 2021. Recent dynamic vehicle routing problems: A survey. Comput. Ind. Eng. 160, 107604.

Ritzinger, U., Puchinger, J., Hartl, R.F., 2016. A survey on dynamic and stochastic vehicle routing problems. Int. J. Prod. Res. 54 (1), 215–231.

Sarasola, B., Doerner, K.F., Schmid, V., Alba, E., 2016. Variable neighborhood search for the stochastic and dynamic vehicle routing problem. Ann. Oper. Res. (ISSN: 1572-9338) 236 (2), 425–461. http://dx.doi.org/10.1007/s10479-015-1949-7.

Sayarshad, H.R., Chow, J.Y.J., 2015. A scalable non-myopic dynamic dial-a-ride and pricing problem. Transp. Res. Part B: Methodol. (ISSN: 0191-2615) 81, 539–554. http://dx.doi.org/10.1016/j.trb.2015.06.008.

Sayarshad, H.R., Oliver Gao, H., 2018. A scalable non-myopic dynamic dial-a-ride and pricing problem for competitive on-demand mobility systems. Transp. Res. Part C: Emerg. Technol. (ISSN: 0968-090X) 91, 192–208. http://dx.doi.org/10.1016/j.trc.2018.04.007.

Schilde, M., Doerner, K.F., Hartl, R.F., 2014. Integrating stochastic time-dependent travel speed in solution methods for the dynamic dial-a-ride problem. European J. Oper. Res. (ISSN: 0377-2217) 238 (1), 18–30. http://dx.doi.org/10.1016/j.ejor.2014.03.005.

Schyns, M., 2015. An ant colony system for responsive dynamic vehicle routing. European J. Oper. Res. (ISSN: 0377-2217) 245 (3), 704–718. http://dx.doi.org/10.1016/j.ejor.2015.04.009.

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

Soeffker, N., Ulmer, M.W., Mattfeld, D.C., 2022. Stochastic dynamic vehicle routing in the light of prescriptive analytics: A review. European J. Oper. Res. 298 (3), 801–820.

Soeffker, N., Ulmer, M.W., Mattfeld, D.C., 2024. Balancing resources for dynamic vehicle routing with stochastic customer requests. OR Spectrum 1–43.

Srour, F.J., Agatz, N., Oppen, J., 2018. Strategies for handling temporal uncertainty in pickup and delivery problems with time windows. Transp. Sci. 52 (1), 3–19.

Steever, Z., Karwan, M., Murray, C., 2019. Dynamic courier routing for a food delivery service. Comput. Oper. Res. (ISSN: 0305-0548) 107, 173–188. http://dx.doi.org/10.1016/j.cor.2019.03.008.

Tafreshian, A., Abdolmaleki, M., Masoud, N., Wang, H., 2021. Proactive shuttle dispatching in large-scale dynamic dial-a-ride systems. Transp. Res. Part B: Methodol. (ISSN: 0191-2615) 150, 227–259. http://dx.doi.org/10.1016/j.trb.2021.06.002.

Tirado, G., Hvattum, L.M., 2017a. Determining departure times in dynamic and stochastic maritime routing and scheduling problems. Flex. Serv. Manuf. J. (ISSN: 1936-6590) 29 (3), 553–571. http://dx.doi.org/10.1007/s10696-016-9242-x.

Tirado, G., Hvattum, L.M., 2017b. Improved solutions to dynamic and stochastic maritime pick-up and delivery problems using local search. Ann. Oper. Res. (ISSN: 1572-9338) 253 (2), 825–843. http://dx.doi.org/10.1007/s10479-016-2177-5.

Toth, P., Vigo, D., 2002. The Vehicle Routing Problem. SIAM.

Towers, M., Kwiatkowski, A., Terry, J., Balis, J.U., De Cola, G., Deleu, T., Goulao, M., Kallinteris, A., Krimmel, M., KG, A., et al., 2024. Gymnasium: A standard interface for reinforcement learning environments. arXiv preprint arXiv:2407.17032.

Ulmer, M.W., 2019. Anticipation versus reactive reoptimization for dynamic vehicle routing with stochastic requests. Networks (ISSN: 1097-0037) 73 (3), 277–291. http://dx.doi.org/10.1002/net.21861.

Ulmer, M.W., 2020. Dynamic pricing and routing for same-day delivery. Transp. Sci. (ISSN: 0041-1655) 54 (4), 1016–1033. http://dx.doi.org/10.1287/trsc.2019.0958, Publisher: INFORMS.

Ulmer, M.W., Goodson, J.C., Mattfeld, D.C., Hennig, M., 2019a. Offline–online approximate dynamic programming for dynamic vehicle routing with stochastic requests. Transp. Sci. (ISSN: 0041-1655) 53 (1), 185–202. http://dx.doi.org/10.1287/trsc.2017.0767, Publisher: INFORMS.

Ulmer, M.W., Goodson, J.C., Mattfeld, D.C., Thomas, B.W., 2020. On modeling stochastic dynamic vehicle routing problems. EURO J. Transp. Logist. 9 (2), 100008.

Ulmer, M.W., Heilig, L., Voß, S., 2017. On the value and challenge of real-time information in dynamic dispatching of service vehicles. Bus. Inf. Syst. Eng. 59, 161–171.

Ulmer, M.W., Mattfeld, D.C., Köster, F., 2018. Budgeting time for dynamic vehicle routing with stochastic customer requests. Transp. Sci. 52 (1), 20–37.

Ulmer, M.W., Streng, S., 2019. Same-day delivery with pickup stations and autonomous vehicles. Comput. Oper. Res. (ISSN: 0305-0548) 108, 1–19. http://dx.doi.org/10.1016/j.cor.2019.03.017.

Ulmer, M.W., Thomas, B.W., 2018. Same-day delivery with heterogeneous fleets of drones and vehicles. Networks 72 (4), 475–505.

Ulmer, M.W., Thomas, B.W., Campbell, A.M., Woyak, N., 2021. The restaurant meal delivery problem: Dynamic pickup and delivery with deadlines and random ready times. Transp. Sci. 55 (1), 75–100.

Ulmer, M.W., Thomas, B.W., Mattfeld, D.C., 2019b. Preemptive depot returns for dynamic same-day delivery. EURO J. Transp. Logist. 8 (4), 327–361.

van Heeswijk, W.J.A., Mes, M.R.K., Schutten, J.M.J., 2019. The delivery dispatching problem with time windows for urban consolidation centers. Transp. Sci. (ISSN: 0041-1655) 53 (1), 203–221. http://dx.doi.org/10.1287/trsc.2017.0773, Publisher: INFORMS.

Voccia, S.A., Campbell, A.M., Thomas, B.W., 2019. The same-day delivery problem for online purchases. Transp. Sci. 53 (1), 167–184.

Vonolfen, S., Affenzeller, M., 2016. Distribution of waiting time for dynamic pickup and delivery problems. Ann. Oper. Res. (ISSN: 1572-9338) 236 (2), 359–382. http://dx.doi.org/10.1007/s10479-014-1683-6.

Wang, X., Kopfer, H., 2015. Rolling horizon planning for a dynamic collaborative routing problem with full-truckload pickup and delivery requests. Flex. Serv. Manuf. J. (ISSN: 1936-6590) 27 (4), 509–533. http://dx.doi.org/10.1007/s10696-015-9212-8.

Xiang, X., Tian, Y., Zhang, X., Xiao, J., Jin, Y., 2022. A pairwise proximity learning-based ant colony algorithm for dynamic vehicle routing problems. IEEE Trans. Intell. Transp. Syst. (ISSN: 1558-0016) 23 (6), 5275–5286. http://dx.doi.org/10.1109/TITS.2021.3052834, Conference Name: IEEE Transactions on Intelligent Transportation Systems.

Zhang, H., Ge, H., Yang, J., Tong, Y., 2022. Review of vehicle routing problems: Models, classification and solving algorithms. Arch. Comput. Methods Eng. 29 (1), 195–221.

Zhang, J., Luo, K., Florio, A.M., Van Woensel, T., 2023. Solving large-scale dynamic vehicle routing problems with stochastic requests. European J. Oper. Res. (ISSN: 0377-2217) 306 (2), 596–614. http://dx.doi.org/10.1016/j.ejor.2022.07.015.

Zhang, S., Ohlmann, J.W., Thomas, B.W., 2018. Dynamic orienteering on a network of queues. Transp. Sci. http://dx.doi.org/10.1287/trsc.2017.0761, Publisher: INFORMS.

Zhang, J., Van Woensel, T., 2023. Dynamic vehicle routing with random requests: A literature review. Int. J. Prod. Econ. 256, 108751.

Zolfagharinia, H., Haughton, M., 2014. The benefit of advance load information for truckload carriers. Transp. Res. Part E: Logist. Transp. Rev. (ISSN: 1366-5545) 70, 34–54. http://dx.doi.org/10.1016/j.tre.2014.06.012.

Zolfagharinia, H., Haughton, M., 2016. Effective truckload dispatch decision methods with incomplete advance load information. European J. Oper. Res. (ISSN: 0377-2217) 252 (1), 103–121. http://dx.doi.org/10.1016/j.ejor.2016.01.006.