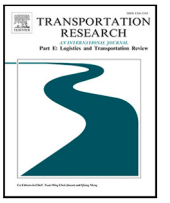
Transportation Research Part E 191 (2024) 103742

Contents lists available at ScienceDirect

Transportation Research Part E

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



Capacity planning in logistics corridors: Deep reinforcement learning for the dynamic stochastic temporal bin packing problem

Amirreza Farahani *, Laura Genga, Albert H. Schrotenboer, Remco Dijkman

Industrial Engineering & Innovation Sciences, Eindhoven Artificial Intelligence Systems Institute, Eindhoven University of Technology, The Netherlands

ARTICLE INFO

Keywords:
Transportation
Logistics
Deep reinforcement learning
Stochastic programming
Logistics corridors
Bin packing
Real-time planning
Continuous-time planning

ABSTRACT

This paper addresses the challenge of managing uncertainty in the daily capacity planning of a terminal in a corridor-based logistics system. Corridor-based logistics systems facilitate the exchange of freight between two distinct regions, usually involving industrial and logistics clusters. In this context, we introduce the dynamic stochastic temporal bin packing problem. It models the assignment of individual containers to carriers' trucks over discrete time units in real-time. We formulate it as a Markov decision process (MDP). Two distinguishing characteristics of our problem are the stochastic nature of the time-dependent availability of containers, i.e., container *delays*, and the continuous-time, or *dynamic*, aspect of the planning, where a container announcement may occur at any time moment during the planning horizon. We introduce an innovative real-time planning algorithm based on Proximal Policy Optimization (PPO), a Deep Reinforcement Learning (DRL) method, to allocate individual containers to eligible carriers in real-time. In addition, we propose some practical heuristics and two novel rolling-horizon batch-planning methods based on (stochastic) mixed-integer programming (MIP), which can be interpreted as computational information relaxation bounds because they delay decision making. The results show that our proposed DRL method outperforms the practical heuristics and effectively scales to larger-sized problems as opposed to the stochastic MIP-based approach, making our DRL method a practically appealing solution.

1. Introduction

Corridor-based integrated logistics systems are key for making transportation networks efficient. They support freight exchange between two main industrial or logistics regions (Crainic et al., 2021) via large terminals that are connected by ample transport modes and transport resources. Corridor-based transportation plays a pivotal role in sustainable logistics as it consolidates freight for full-truckload, long-haul transportation, which allows for the reduction of the environmental impact for the transportation between regions, and the economies of scale also enable eco-friendly practices for the transport in the region's hinterland. Coordinating the freight stream in the corridor is far from trivial, as the arrival of freight is subject to many disturbances and is often delayed. At the same time, transport service providers operating on the corridor want to commit to high customer service by communicating upfront with the customers when transport is going to take place, even before their customers' freight arrives. Thus, it is necessary to anticipate future freight arrival delays while already planning its transport.

* Corresponding author.
E-mail addresses: a.farahani@tue.nl (A. Farahani), l.genga@tue.nl (L. Genga), a.h.schrotenboer@tue.nl (A.H. Schrotenboer), r.m.dijkman@tue.nl (R. Dijkman).

https://doi.org/10.1016/j.tre.2024.103742
Received 4 March 2024; Received in revised form 20 August 2024; Accepted 24 August 2024
Available online 31 August 2024

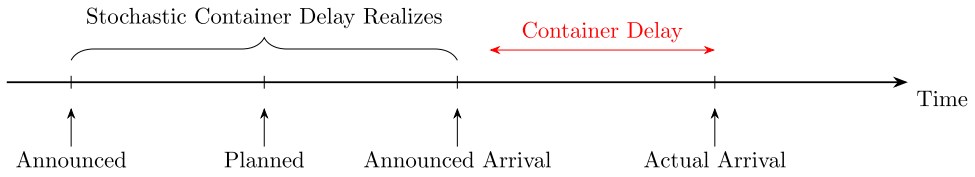


Fig. 1. Visualization of the events associated with planning a single container at a major terminal in a logistics corridor.

Inspired by our collaboration with industry partners, this paper studies the real-time capacity planning of a transport service provider operating from a major terminal in a corridor-based logistics system. The problem asks for dynamically assigning incoming full-truckload containers to a fleet of available trucks. As each container has a different destination, the amount of truck resources needed to fulfill the transport differs. Therefore, it may, for some containers, be cheaper to outsource the transport, because it leaves the transportation service provider more flexibility for using its own trucks.

As visualized in Fig. 1, the typical operations for each container transport are: (i) A few days before a container arrives at the terminal, its upcoming arrival is announced to the terminal; (ii) After the announcement and before the arrival of the container, the transportation service provider communicates *when* the container will be transported; (iii) However, containers might be delayed, making the actual arrival of the container at the terminal stochastic — and this information is typically only known after already communicating *when* the container will be transported. (iv) The container will arrive at the terminal and be planned for transport or outsourced. As a result, it may be necessary to assign trucks to containers before they arrive at the logistics hub, and thus, resources should be reserved before uncertainty (i.e., delays of arriving containers) is fully revealed. How to cost-efficiently assign stochastically arriving containers to an available fleet of trucks to minimize cost is a complex optimization task.

In this paper, we introduce the dynamic stochastic temporal bin packing problem (DSTBPP), which models capacity planning at a terminal in a corridor-based integrated logistics system. The classical Bin Packing Problem (BPP) aims to load a set of items of different weights into the smallest possible number of bins with a given capacity while ensuring the packing of each bin is feasible (Dyckhoff, 1990). In logistics capacity planning, the incoming containers can be represented by items, and each of the periods represents a bin with a capacity equal to the number of available trucks during that period. Whereas many variants of the BPP have been studied in the literature to model logistics capacity planning problems (Perboli et al., 2014; Baldi et al., 2019), our DSTBPP is novel by considering the real-time interplay between container announcement, delay, and the need to plan before the actual delay is known. To the best of the authors' knowledge, we are the first to consider this setting. While we specifically look at DSTBPP as a model for solving capacity planning in logistics corridors, it can be used for other applications as well. Examples include allocating picking orders to resources in warehouses, where picking orders arrive dynamically and picking time is stochastic, allocating risk assessment tasks of different types to employees at a bank, and allocating containers to barges in hinterland transport (see, e.g., Beeks et al., 2022; Gumuskaya et al., 2021)

The DSTBPP falls within the area of dynamic combinatorial optimization, for which the solution typically is a decision policy. Reinforcement Learning (RL) theoretically offers an ideal solution method for dynamic combinatorial optimization and, therefore, DSTBPP, because these problems can be naturally modeled as Markov decision processes (MDPs) as discussed by Powell (2019, 2021). Moreover, learning architectures like neural networks in deep RL can be trained offline to recognize patterns of dynamism and stochasticity in the planning environment, thus enabling the quick generation of complex decision policies (Hildebrandt et al., 2023).

We model the DSTBPP as a MDP. But for realistically sized instances, we face the three curses of dimensionality in states, actions, and transitions. We, therefore, propose two tailored solution approaches to solve the DSTBPP. First, we consider a DRL approach to plan the announced (and re-plan delayed) containers in real-time. This approach is motivated by observations at our industry partner, for which it is essential that containers can be planned in real time to align related downstream and upstream planning processes. Moreover, DRL offers adaptability to dynamic environments, learning optimal policies through experience. Its capability to handle stochasticity makes it particularly well-suited for logistics scenarios with uncertain events (Tseng et al., 2023), such as varying demand, delay, and transit times as in the DSTBB. Recent advancements have demonstrated its efficacy in various logistics applications. The DRL approach that we propose is based on PPO (Schulman et al., 2017), a powerful DRL method that has shown promising results in various fields and has demonstrated notable success in navigating stochastic environments (Heess et al., 2017). This method is able to learn policies for allocating individual containers to eligible trucks while the plan is being executed. This real-time planning is advantageous in practice, as it reduces the need for batching containers (and thus unnecessary waiting) and is appealing for providing decision support.

Second, we propose a rolling-horizon two-stage Stochastic Programming (SP) approach in which we can plan a batch of announced (and to be re-planned) containers at once. This second approach quantifies uncertainty by providing explicit probabilistic scenarios. It can also be interpreted as a computational information relaxation bound as we delay decision-making until a batch of containers has to be planned, compared to the real-time planning fashion of DRL. Also note that the rolling horizon SP approach requires full distributional knowledge of uncertainty, while DRL is flexible and does not require this. We also consider an oracle solver based on a perfect information MIP formulation.

Our results demonstrate the efficacy of our PPO-based algorithm. Additionally, our comparison with a state-of-the-art DQN approach further confirms that PPO is an excellent choice for a DRL-based algorithm. It outperforms several practical heuristics that

are typically used in corridor-based logistics systems, and inspired upon consultation with our industry partners. Our PPO-based algorithm scales effectively to larger-sized problems compared to the rolling-horizon SP approach. Our PPO-based algorithm achieves an average cost reduction within 4.7% to 5.41% and utilizes a 5.49% to 11.11% greater amount of own truck capacity compared to practical heuristics on several problem instances – needing a lower number of costly outsourcing options. The information-relaxation stochastic programming approach only achieves 5.59% to 8.21% better solutions – at the expense of delaying all decision-making to the end of the day, which is according to our partners from practice not preferred, and at the expense of a much higher computational complexity while executing the planning.

Summarizing, we make the following scientific contributions:

- We introduce the DSTBPP to address the challenges in logistics capacity planning at a terminal in a major logistics corridor. The novelty stems from the fact that containers need to be planned before they arrive in the system – to be able to inform the customer about their arrival – while being subject to a stochastic delay between announcement and arrival.
- We propose two tailored algorithms for solving the DSTBPP. First, we consider a novel planning approach based on PPO, a popular DRL method, to assign containers in real-time, which eradicates the need for online computationally expensive alternatives such as stochastic programming. Second, we develop a rolling-horizon two-stage stochastic programming approach that, besides being used as a benchmark, also offers value by itself. It is a potential approach towards capacity planning at a terminal in a logistics corridor in case decision-making can be delayed towards the end of each planning period. To the best of the author's knowledge, no such batch-planning approaches exist yet in the literature for solving the DSTBPP.
- We show that our PPO-based algorithm learns high-quality policies, outperforms practical heuristics, and only performs marginally worse compared to the two-stage stochastic programming approaches. This latter approach is, however, much more computationally expensive and requires orders to be planned in batches which has drawbacks in practice. In such cases dependent down- and upstream planning processes cannot be managed efficiently.

The remainder of the paper is organized as follows. Section 2 introduces the relevant literature. Section 3 provides a formal definition of the new Bin Packing Problem variant. Section 4 introduces our methods. Section 5 discusses our experimental evaluation. Section 6 draws conclusions.

2. Literature review

The DSTBB is a novel variant of the BPP. We, therefore, focus our review on variants of the BPP often used for modeling capacity planning in logistics. Table 1 provides such an overview, where we categorize the problems on factors motivated by Perboli et al. (2018), Crainic et al. (2021). We briefly discuss each of these factors in the following:

1. The number of 'dimensions' considered in the problem, which refers to the physical and handling characteristics of items or bins, can be 'single' or 'multiple'. A 'single' dimension means that only one characteristic, such as size, is considered, while 'multiple' dimensions mean that more than one characteristic, such as shapes and volume, is considered.
2. The 'bin cost' describes how costs are associated with each bin in the problem. Costs can either be 'fixed' or 'variable'. 'Fixed' costs are the same for all items and bins, while costs can also be 'variable' for each bin and item, for example, due to cost differences between own trucks and spot market options.
3. The 'bin size' describes the size or capacity of the bins and can either be 'identical', 'equal/proportional', or 'variable'. 'Identical' means that the size is the same for all bins, while 'equal/proportional' means that the size is proportional to the cost of the bin. 'Variable' means that the size may be different for each bin.
4. The 'commodity' refers to the number of different item types. There can either be a 'single' commodity/item type or 'multi'-commodity/item types.
5. The representation of 'time' can be categorized as: 'single period', 'multi-period', or 'continuous'. A 'single period' problem does not explicitly consider time, and has no time-related attributes for bins and items. All events and decisions occur simultaneously. On the other hand, a 'multi-period' problem discretizes the decision-planning horizon into a number of time periods, with all time-related events (e.g., the starting of a bin availability interval) associated with a specific time period. Finally, if an event can occur at any time during the decision-planning horizon, the planning problem is considered 'continuous'.
6. The 'stochasticity' factor represents the uncertainty aspects considered by the problem. In the literature, we found a stochastic number of bins, as well as stochastic costs, demand, and arrival time (i.e. the arrival can be delayed compared to the estimated time of arrival).

The Variable Size Bin Packing Problem (VSBPP) for logistics capacity planning was initially introduced in a study by Wäscher et al. (2007). This problem class deals with bins of different sizes and multiple physical characteristics, including single costs that are either equivalent to their sizes or strictly proportional to them. The items in this class are of a single type, and the problem is confined to a single period, meaning that all events and decisions occur simultaneously for a single planning period, and no stochasticity is taken into account.

To address practical concerns, the Variable Cost and Size Bin Packing Problem (VCSBPP) was introduced in several studies by Monaci (2003), Kang and Park (2003), Pisinger and Sigurd (2005), Alves and De Carvalho (2007), Crainic et al. (2011). This problem class disconnects the definition of the costs of the bins from their sizes and considers a variable cost as well as size. The remaining factors in this problem are the same as the Variable Size Bin Packing class.

Table 1
Literature on transportation and logistics capacity planning via a bin packing formulation.

Reference	Dimension	Cost	Size	Commodity	Time	Stochasticity
Wäscher et al. (2007)	multi	fixed	variable	single	single	–
Monaci (2003), Kang and Park (2003), Pisinger and Sigurd (2005), Alves and De Carvalho (2007), Crainic et al. (2011)	multi	variable	variable	single	single	–
Perboli et al. (2012, 2014)	multi	variable	variable	multi	single	costs and profits
Crainic et al. (2014, 2016), Perboli et al. (2018)	multi	variable	variable	single	single	number of bins, costs, demand
Baldi et al. (2012, 2014, 2019)	multi	variable	variable	multi	single	–
Dell'Amico et al. (2020)	single	fixed	identical	single	multi	–
Crainic et al. (2021)	single	variable	variable	single	single & multi	–
This paper	**single**	**variable**	**identical**	**single**	**multi & continuous**	**future demand, delay in arrival**

Perboli et al. (2012) and Perboli et al. (2014) introduced the concept of stochastic costs and profits, with a focus on the stochasticity of real-world problems. They proposed a stochastic variant of the VCSBPP with multiple item types.

Concurrently, (Crainic et al., 2014, 2016) proposed a stochastic extension of the VCSBPP, introduced earlier in a study by Crainic et al. (2011), known as the Stochastic Variable Cost and Size Bin Packing Problem (SVCSBPP). This stochastic variant integrates uncertainty regarding the future item and bin characteristics into the current decision-making process for bin packing. While the VCSBPP considers bins with varying costs and capacities, the SVCSBPP introduces uncertainty in both the bins, such as the number of available bins and their costs, and the demand, such as the number of items and their volumes. Perboli et al. (2018) further considers the Stochastic VCSBPP with loss availability capacity, which explicitly takes into account the actual volumes of the bins.

Another bin packing problem class is the Generalized Bin Packing Problem (GBPP), which extends the classic Bin Packing Problem with multiple item types. It includes two categories of items: compulsory items that must be loaded into bins and non-compulsory items that can either be loaded for an additional profit or left behind at no cost. Baldi et al. (2012, 2014) proposed this problem class, and Baldi et al. (2019) extended it further by introducing bin-dependent profits, where loading items into specific bins can earn additional profit, thereby maximizing the total net revenue. This variant is known as the GBPP with Bin-dependent Item Profit and introduces a new aspect to the problem in a single-period BPP model. However, none of the variants of the GBPP take the stochastic nature of the problem into account.

Recently, the research community has begun to focus on bin-packing problems that consider temporal aspects. Dell'Amico et al. (2020) introduced the Temporal Bin Packing Problem (TBPP), which is a generalization of the standard BPP with single bin costs and sizes in which each item must be packed within a specific time interval or window. This version takes identical bins into account and can be classified as Time Windows on Item assignment.

In a recent study, Crainic et al. (2021) highlighted that the costs associated with assigning items to bins are not solely determined by the physical characteristics of the items and bins. Instead, these costs are heavily influenced by the timing of the assignment and dispatch to ensure prompt delivery of items. The existing literature often overlooks the time-dependent availability of items and bins with various types of assignments. To address this gap, the authors extended the work of Dell'Amico et al. (2020) to include single and multi-period BPP with variable bin costs and sizes, as well as item-to-bin assignment costs that account for the time-dependent availability of items and bins with different assignment possibilities. However, both of these temporal variants of bin packing problems do not consider any stochastic factors.

The literature review presented in this paper highlights the lack of attention given to two major problem aspects that are often encountered in logistics and transportation applications. First, the stochastic nature of the time-dependent availability of both items and bins of various types is ignored. Second, the continuous time aspect, where an event may occur at any time moment during the decision-planning horizon, is ignored. Neglecting these two aspects can have a significant impact on the feasibility of plans, as discussed in Section 3.

To bridge this gap, we introduce the *DSTBPP*. Our approach is a stochastic continuous time extension of the multi-period TBPP introduced by Crainic et al. (2021). Section 3 explains the details of the proposed problem.

It is noteworthy to highlight another area of research dedicated to capacity planning, which approaches the issue through the lens of dispatching and scheduling. However, the focus on specific contexts within these studies highlights the need for further

exploration of dynamic stochastic environments and may not directly address the specific problem settings introduced in this paper. Several studies address the complexities of dynamic service network design (DSND). For instance, (Dall'Orto et al., 2006), focuses on the stochastic aspects in a single-terminal dispatching services scenario. They propose a time-dependent, stochastic formulation with a dynamic programming solution approach optimized over a planning horizon. Within shipping warehouses domain, (Tadumadze and Emde, 2022) investigate the operational planning problem of loading and scheduling outbound trucks. Their research, particularly relevant for just-in-time or just-in-sequence delivery systems, formulates a mixed-integer linear programming model to optimize resource allocation (logistics workers, dock doors) while respecting time windows for loading and scheduling trucks at docks. Zolfagharinia and Haughton (2016) identify limitations of static dispatching rules for long-haul transportation, particularly those that do not account for driver and truck return needs. Their work proposes a two-index MIP model suitable for dynamic contexts with rolling horizons. Additionally, they introduce a deadhead coefficient policy for improved carrier profits. Durbin and Hoffman (2008) contribute a seminal work in ready-mixed concrete (RMC) delivery optimization. They formulate the RMC delivery problem as a time–space network, where nodes represent truck locations and arcs signify deliveries. This framework allows for optimizing RMC delivery operations by considering factors such as delivery locations, loading statuses, and time constraints. While this work showcases a network optimization approach, it may not be directly applicable to logistics corridors due to the specific characteristics of RMC delivery. Looking beyond individual companies, (Schrotenboer et al., 2020) propose a share-first-plan-second policy to encourage collaboration among transportation firms. This policy prioritizes creating a cyclic schedule for shared transportation before real-time shipment assignment. This approach promotes collaboration without extensive planning efforts and demonstrates comparable performance to full collaboration in modal split and fill rates. In conclusion, while the existing literature offers valuable insights into dispatching and scheduling for different complex capacity planning problems in transportation, it may not directly translate to the specific characteristics of the DSTBPP introduced in this paper.

The majority of exact solutions or mathematical programming methods for BPP in the literature rely on decomposition techniques (Alves and De Carvalho, 2007; Casazza and Ceselli, 2016) or reformulations solved using commercial MIP solver software (Correia et al., 2008). However, because they are limited to small size problems, the computational effort required by these methods makes them difficult to employ in real-life (Dolan and Moré, 2002). As a result, heuristic solution approaches are developed, which allow for the discovery of good solutions to larger-scale problems within less computational time.

Recent advancements in the field of Stochastic Dynamic Combinatorial Optimization, particularly in addressing Stochastic Dynamic Vehicle Routing Problems (SDVRPs), underscore the formidable challenge of making timely routing decisions amid uncertainty (Hildebrandt et al., 2023). A recent survey indicates that traditional Reinforcement Learning (RL) methods have encountered difficulties in handling the combinatorial nature of action spaces in SDVRPs, often resorting to limiting action spaces or relying heavily on heuristics. This observation emphasizes the necessity for comprehensive RL strategies capable of effectively managing both the complexity of combinations and the uncertainties of the future. Leveraging these insights, we propose the utilization of DRL for DSTBPP, with the aim of constructing a robust framework. DRL offers promise as neural networks can be trained offline to swiftly generate intricate decision policies that anticipate future demand and adeptly adjust to unforeseeable events in the logistics corridors' capacity planning. Nevertheless, existing literature extensively covers heuristics such as First-Fit and Best-Fit, alongside their variants, for solving BPP (Baldi et al., 2012; Crainic et al., 2021).

In a recent study (Bai et al., 2023) explored the latest research trends in Machine Learning (ML) supported modeling and optimization of VRP. Their analysis illustrates how ML can significantly enhance VRP modeling and improve algorithm performance for both online and offline VRP optimizations. Despite the effectiveness of ML methods particularly popular classification techniques like decision trees and evolutionary algorithms such as genetic algorithms, which offer good interpretability for combinatorial optimization problems (COPs), their application in the context of logistics corridor capacity planning is limited due to struggles in handling dynamic and stochastic environments. Decision trees, relying on historical data, are unable to capture real-time fluctuations in logistics demand and supply, potentially leading to suboptimal capacity allocations and operational inefficiencies. Similarly, evolutionary algorithms, known for their iterative nature and computational intensity, may struggle to promptly adjust to evolving optimal solutions in dynamic environments. While they can provide satisfactory solutions with reduced computational resources compared to exact methods, they do not guarantee to achieve global optimality (Gao et al., 2019; Blum and Roli, 2003). Moreover, their effectiveness often depends on parameter settings that vary across different problem instances or configurations. Furthermore, deploying these algorithms typically necessitates expertise in their design and implementation (Bianchi et al., 2009; Farahani et al., 2023). Hence, exploring alternative optimization methodologies tailored to the unique challenges of logistics corridors. Several studies have employed sequential planning methods, particularly Approximate Dynamic Programming (ADP) (e.g., Bikker et al., 2020; Rivera and Mes, 2022; Dall'Orto et al., 2006). ADP tackles problems sharing characteristics with those addressed by our DRL approach, particularly their stochastic and dynamic nature, and their focus on sequential decision-making. This indicates a significant overlap in their applicability, suggesting that where DRL is applicable, ADP can be used as well. However, because DRL leverages neural networks, typically it can learn more complex patterns and consequently take better decisions in stochastic and dynamic settings (Hildebrandt et al., 2023). ADP, at the same time, has the benefit that policies that are generated by it are better explainable analytically. Since the primary focus of this paper was on finding a policy that performs well in a specific problem setting, DRL was chosen as a solution method over ADP, but we expect ADP to work for this problem setting as well.

In this study, we present two distinct approaches to tackle this challenge: a real-time DRL-based method employing PPO and a rolling-horizon two-stage stochastic programming method. Both these approaches are able to explore the solution space more exhaustively than heuristics and for that reason have demonstrated impressive performance compared to heuristics in dealing with complex and dynamic problems to address the DSTBPP. To demonstrate the benefits of our approaches over heuristic approaches we will compare the performance of our approaches to that of heuristics.

Table 2
System elements and parameters.

Sets	
L	set of containers
D	set of days in the planning problem
Decision variables	
$x_i^d \in \{0,1\}$	own truck starts to transport container $i \in L$ on day $d \in D$
$y_i \in \{0,1\}$	container $i \in L$ is transported by charter truck
$z_i \in \{0,1\}$	planning of container $i \in L$ is postponed to the next day
Parameters	
e_i	estimated day of availability of container $i \in L$
l_i	latest day of arrival of container $i \in L$
del_i	number of days required for delivery of container $i \in L$
cap_d	the capacity on day $d \in D$
C^{OWN}	our truck cost
C^{CHARTER}	charter truck cost
$late_i$	number of days container $i \in L$ arrives late

3. Problem description

This section first provides a system description of the corridor capacity planning problem as a DSTBPP, introducing the main system elements and decisions. Subsequently, we formulate the DSTBPP as a MDP.

3.1. System description

We consider a single home terminal as a depot that must efficiently allocate a stochastic supply of containers to a fleet of available trucks. In the following, we first introduce the main elements of the system. Afterward, we discuss the sequence of events and the decisions to be made.

Table 2 presents an overview of the system's elements and parameters. We consider a continuous time horizon $\mathcal{T} := [0, T]$. Each time point $t \in \mathcal{T}$ has an associated discretized time period $d \in D$. Without loss of generality, we refer to the discretized time as days, but it can also be set to weeks, parts of days, or hours, depending on practical considerations. We consider a set of stochastically arriving containers that we collectively denote by L. The set L is thus only fully known to the decision maker at the end of the time horizon. Each container $i \in L$ must be transported from the home terminal to a designated terminal in a different region. Each container i is characterized by three temporal attributes: (1) the estimated day e_i at which the container arrives at the terminal (subject to delay) and is available for transport, (2) the delivery due date l_i at which the container must be at the designated terminal at its destination location in a different region, and (3) the required number of days del_i for delivery of the container. Motivated by the daily practice at our industry partner, we assume that at the start of each day, the number of new containers that will arrive in the system is known, either due to a good forecast procedure or by expert knowledge available in the planning department. This means that for each container $i \in L$, on the day on which it was estimated to arrive, the number of days that it will be delayed $late_i$ is revealed.

We like to stress that in our model we consider the containers to be planned one-by-one. As we assume that at the start of each day the number of new containers is known, we have some freedom to select the order in which we plan. Consequently, we need to make a choice on which container to plan first, before applying the DRL algorithm to select the way in which to plan the container. We consider two approaches to select the container to plan first: FIFO and EDF. The first-in-first-out (FIFO) approach operates in a fully real-time/online manner, where no additional information about other containers is required. As soon as an individual container is announced to the system, it can be planned immediately. This means that we order containers based upon their arrival time to the system. On the other hand, the earliest due date first (EDF) approach requires information about the due date of all containers arriving in a day, and consequently orders them based on their due date. In the evaluation section, we evaluate the DRL algorithm both in combination with the FIFO heuristic and in combination with the EDF heuristic.

To transport the containers, we can either make use of the capacity in a so-called pool of transportation service providers, which we call 'own trucks', or we hire a truck from the spot market which we call 'charter trucks'. The own trucks impose a capacity cap_d for each day d in the planning horizon. This implies that if we plan a container i that takes del_i to transport on day d, then we reduce the number of available trucks from days d to $d + 2 \times del_i - 1$ by 1. Using an own truck comes at a cost $C^{\text{OWN}} > 0$. The charter trucks are assumed to have unlimited capacity but come at a higher cost $C^{\text{CHART}} > C^{\text{OWN}}$.

Events in our system happen in real-time. Thus, at some continuous time point $t \in \mathcal{T}$ with associated day $d \in D$, a container i is first announced to the system. At that moment in time, an expected earliest day of availability $e_i \in D$ is shared. Note $e_i > d$ and subject to potential delay. The decision-maker is required to decide on the planning of the container transport k days before e_i. The reason for this is the service contract that the decision-maker has with its clients, where it must communicate the actual departure and arrival of the transport at least k days in advance. Note this might imply that we decide on the planning more than k days before e_i, as it is subject to delay. Since it is typically not beneficial to decide upon the planning of a container more than k days before the expected arrival time e_i, the decision maker is allowed to postpone the planning decision to the next day. Note that the

actual planning should respect the deadline for delivery of the container. Due to postponing planning decisions, we also consider events associated with replanning the postponed containers. We detail this in the next subsection.

The possible decisions about the assignment of transport options to containers are represented by three binary variables x_i^d, y_i, and z_i. x_i^d represents whether an own truck transports container $i \in L$, starting on the day $d \in D$. Here, y_i represents whether container i is transported by a charter truck. As charter trucks are assumed to be always available and uncapacitated, it is not necessary to encode them with a day of departure. Furthermore, z_i represents whether the planning of container i is postponed to the next day. The objective is to minimize the expected total cost of the system, while ensuring the capacity of the own trucks is respected and all containers are transported to their destination on time (detailed constraints are given in Section 3.2).

3.2. Markov decision process

We formulated the problem as a MDP in line with the framework introduced by Powell (2019, 2021).

State variable: The state $s_t \in S_t$ of the system at decision step t is defined by the tuple $s_t = (d, \nu, Q, \hat{L}, \mho, \hbar)$. Here, $d \in D$ is the current day, $\nu \in L$ is the container that should be planned, Q is the vector of capacities cap_u for all future days $u \in \{d+1, \dots, D\}$, and $\hat{L} \subset L$ is the set of all announced yet unplanned containers. Finally, $\mho$ is the set of postponed containers, and $\hbar$ are the assigned containers. Note we use ν to stress that this is the container to be planned while we reserve index i for any possible container $i \in L$.

Decision variable: Let $a_t \in A_t(S_t)$ be the decision associated with container ν at decision step t. The decision a_t is described by binary decision variables $x_\nu^u, u \in \{d+1, \dots, D\}$, y_ν, and z_ν for container ν. Here $x_\nu^u = 1$, if container ν is assigned to an own truck at day $u > d$ and equals 0 otherwise, y_ν equals 1 if a charter truck is used for transporting container ν and equals 0 otherwise, and z_ν, equals 1 if we postpone deciding on the planning to the next day and equals 0 otherwise. Note that within the RL framework, the decision variables correspond to actions.

Several constraints limit the feasibility of decision a_t. First, the container ν is either scheduled, postponed, or outsourced, implying that $y_\nu + z_\nu + \sum_{u \in \{d+1, d+2, \dots, D\}} x_\nu^u = 1$. Second, we cannot plan a container before its scheduled arrival time, so $x_\nu^u = 0$ for all $u \in \{d+1, \dots, e_\nu - 1\}$. Similarly, we cannot plan a container on an own truck if we cannot meet the deadline, so $x_\nu^u = 0$ for all $u \in \{l_\nu - del_\nu, \dots, D\}$. Third, we can only postpone if $d \leq e_\nu - k$, otherwise we let $z_\nu = 0$ by construction. This implies that if $d = e_\nu - k$, we have to plan the container on an own truck or outsource it. Finally, x_ν^u can only equal 1 if $cap_i \geq 1$ for all $i \in \{u, \dots, u + 2 \times del_i - 1\}$.

Exogenous information: Let W_{t+1} denote the exogenous information revealed between transitioning between states t and $t+1$. We consider two separate cases. First, if we transition to assigning the next container from the set $\hat{L}$ within the same day, the exogenous information is the next container to be scheduled according to the dispatching heuristics, as explained before.

Second, if we transition between two days, we consider two types of exogenous information: new order arrival announcements and order delays. The total number of containers that are announced each day is a random variable denoted by N. Recall that each container $i \in L$ is characterized by the estimated day of availability at the terminal e_i, the latest day of arrival at the destination l_i, and the required number of days for delivery to the destination del_i.

The order delays are represented by the number of days $late_i$ the container is delayed compared to the original announcement e_i. This is revealed for all containers that were supposed to arrive on the new day (i.e., for which $e_i = d+1$).

Transition function: We denote the transition function by $s_{t+1} = S^M(s_t, a_t, W_{t+1})$. The transitions are a multi-step process defined by two types of transitions: the transition after assigning each container ν except the last container of the day, and the transition after assigning the last container of the day and transitioning to the next planning day. Here, the last container of the day is defined as the container that is planned at planning day d, and there are no unplanned containers left at $\hat{L}$ except those that are postponed (i.e., $\mho$) to the next planning day $d+1$.

During the transition after assigning container ν from L except the last container of the day, we have three possible scenarios.

- If the selected truck for container ν is an own truck, (i.e., $x_\nu^u = 1$ for some $u > d$, we update the capacity based on the selected day of transport, i.e.,

$$cap_i' = cap_i - 1, \qquad \forall i \in \{u, \dots, u + (del_\nu \times 2) - 1\}, \tag{1}$$

  where Q' is the vector of cap_i'. Subsequently, we remove the selected order ν from the list of unplanned orders, i.e., $L' = L \backslash \{\nu\}$, add the planned order ν to the tracking list of planned orders, i.e., $\hbar' = \hbar \cup \{\nu\}$, and set $\mho' = \mho$.
- If the selected truck is a charter truck, i.e., $y_\nu = 1$, we remove the selected order ν from the list of unplanned orders, i.e., $\hat{L}' = L \backslash \{\nu\}$. Furthermore, we set $Q' = Q$, $\mho' = \mho$, and $\hbar' = \hbar$.
- If the selected action is to postpone the assignment of the order to a truck, i.e., $z_\nu = 1$, we set $\mho' = \mho \cup \{\nu\}$, $Q' = Q$, $\hbar' = \hbar$, and $\hat{L}' = \hat{L} \backslash \{\nu\}$.

Finally, in any of these cases, we need to determine the next container ν' from $\hat{L}$ to be planned in the upcoming decision epoch. Here, we let this be based on a so-called priority dispatching heuristic. It is worth noting that there are numerous priority dispatching heuristics available in the literature, such as First-In-First-Out (FIFO), Shortest Processing Time (SPT), and Earliest Due Date First (EDF), which are commonly used in this field. We use and compare two of these dispatch heuristics, namely FIFO and EDF. Under FIFO, the next container ν' is simply the container in $\hat{L}$ that arrived earliest in the system, while under EDF, it is the container with the earliest due date in $\hat{L}$. The state in which we transition is then $s_{t+1} = (d, \nu', Q', \hat{L}', \mho', \hbar')$.

After assigning the last container of the day, assuming $d < D$, a different transition takes place. Now, delays and new announcements are revealed to the system. In this case, $s_{t+1} = (d+1, \emptyset, Q'', \hat{L}'', \emptyset, \hbar'')$, and is defined as follows.

We first check the assigned orders $\hbar$ to determine the earliest day of availability of each assigned order $i \in \hbar$. If the expected day of arrival of any of these containers is the next day and it has a delay (i.e., $e_i = d + 1 \wedge late_i \geq 1$), we need to check if this delayed container can still reach the departure day of their assigned trucks. If they can, we do not need to make any updates to our plan. However, if they arrive after the departure day of the assigned truck, we need to cancel their assignment, release their capacity, and postpone these orders to the next planning day for re-planning or re-assignment by adding the container to the unplanned orders list L'. This is how we obtain $\hbar''$ from $\hbar$ and how we obtain Q'' from Q. Besides the aforementioned orders, the set L'' is enlarged by $\mho$ and the newly revealed orders on day $d + 1$.

The transition process occurs iteratively after the assignment of each container until all containers within the planning horizon have been planned. Once all containers have been planned, there are no unplanned containers left, and the planning process is complete.

Objective function: We let the reward r_t of taking action a_t given the information in state s_t be the negative cost for each associated transport option. That is, we set $R(s_k, a_k) = -\left(x_v^u C^{\text{OWN}} + y_v C^{\text{CHARTER}}\right)$. A solution to the DSTBPP is given by a decision policy $\pi \in \Pi$ and a decision rule $X^\pi : S_t \rightarrow \mathcal{A}_t(S_t)$. Note that $a_t = X^\pi(s_t)$. The objective is then to find a policy that maximizes the expected total reward:

$$\max_{\pi \in \Pi} \mathbb{E}\left[\sum_{t \in \mathcal{T}} R(s_t, X^\pi(s_t)) \mid s_0\right], \tag{2}$$

where $s_{t+1} = S^M(s_t, X^\pi(s_t), W_{t+1})$.

4. Methods

We present two distinct approaches for solving the DSTBPP. The first approach is a DRL based approach called PPO. This method plans containers in real-time either based on the FIFO dispatching heuristic or on the end-of-the-day EDF dispatching heuristic. The second approach is a heuristic rolling-horizon batch-planning method that utilizes Two-Stage Stochastic Mixed-Integer Linear Programming. This method works under the assumption that delayed decision-making for each container is allowed, which is not always desirable in a practical setting. Both methods are explained in detail in the following sections.

4.1. Approach 1: Proximal policy optimization

PPO is a real-time planning algorithm based on DRL. By leveraging the power of Deep Neural Networks and Reinforcement Learning, our method enables agents to learn and make sequential decisions in complex and uncertain environments. At each time step t, the agent observes (a subset of) the current state s_t of the environment, selects an action a_t to assign an individual container v to a transportation option based on a policy function, receives a reward signal r_t, and transitions to the next state s_{t+1}. The goal of the agent is to maximize the expected cumulative reward over a sequence of actions (Sutton and Barto, 2018). In simpler terms, the agent aims to minimize the overall costs. Fig. 2 shows an overview on the Real-time planning approach via DRL. Note that the details of the transition process are already discussed in Section 3.2. It is common to feed the model with observations of the state at each time step o_t rather than providing the model with complete state information s_t. We define this observation o_t as the capacities of the planning horizon, the information regarding the container to be planned, and the size of $\hat{L}$. In this approach *Observation Manager* transforms a state s_t into an observation o_t. This transformation maps the information from the state into an observable format o_t that the agent can interpret. This approach is adopted to simplify the neural network architecture and reduce the dimensionality of the input space, while choosing observations that contain sufficient relevant information for the agent to make decisions and learn high-quality policies. By focusing the observations on the most salient aspects of the state, the model can potentially learn more efficiently and generalize better to unseen situations.

The PPO algorithm is a popular Policy Gradient Actor–Critic DRL algorithm. PPO combines trust region policy optimization and clipped surrogate objective methods to ensure good sample efficiency and stability. The algorithm updates policy parameters using a clipped surrogate objective function. At each time step t, the clipped surrogate objective $L_{\text{clip}}(\theta_t)$ is defined as:

$$L_{\text{clip}}(\theta_t) = \mathbb{E}t\left[\min\left(\frac{\pi_{\text{new}}(a_t|s_t)}{\pi_{\text{old}}(a_t|s_t)} A(s_t, a_t), \text{clip}\left(\frac{\pi_{\text{new}}(a_t|s_t)}{\pi_{\text{old}}(a_t|s_t)}, 1-\epsilon, 1+\epsilon\right) A(s_t, a_t)\right)\right] \tag{3}$$

Here, θ_t is the policy parameter vector at time step t, and $\pi_{\text{new}}(a_t|s_t)$ and $\pi_{\text{old}}(a_t|s_t)$ represent the probability of selecting an action a_t in state s_t under the new and old policies, respectively. The ratio $\frac{\pi_{\text{new}}(a|s)}{\pi_{\text{old}}(a|s)}$ compares the probabilities of selecting an action under the new and old policies. The advantage function $A(s_t, a_t)$ measures how much better an action a_t is than the average action in state s_t, and ϵ is a hyperparameter that controls the size of the clipping interval. The expectation is taken over the trajectory starting from time step t.

The clipped surrogate objective encourages the new policy to improve while limiting the change in policy that can occur at each iteration (Schulman et al., 2017). The PPO algorithm also uses a value function baseline to reduce the variance of the gradient estimates. At each time step t, the advantage function is calculated as

$$A(s_t, a_t) = Q(s_t, a_t) - V(s_t). \tag{4}$$

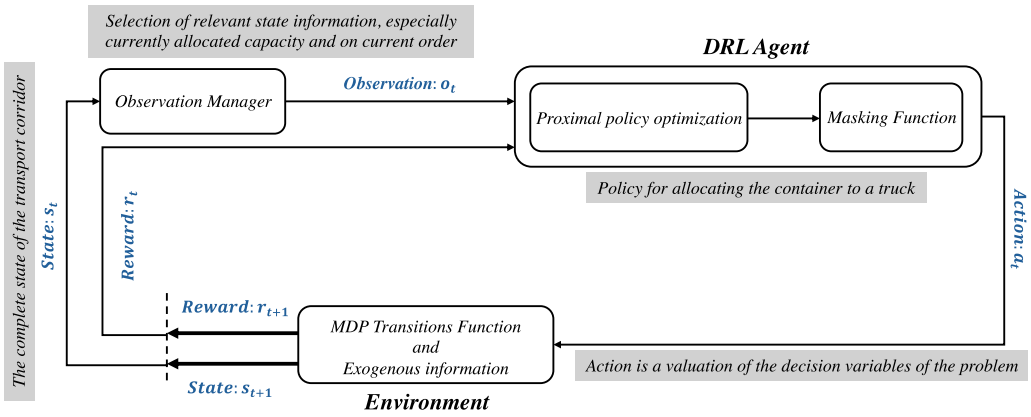


Fig. 2. Overview on the Real-time planning approach via DRL.

Here, $Q(s_t, a_t)$ is the action-value function, which represents the expected cumulative reward from a given state–action pair (s_t, a_t) under a policy, and $V(s_t)$ is the value function, which represents the expected cumulative reward from a given state s_t under a policy function π:

$$Q(s_t, a_t) = \mathbb{E}t[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t, a_t] \tag{5}$$

$$V(s_t) = \mathbb{E}t[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t] \tag{6}$$

Here, γ is the discount factor, r_t is the reward received at time step t, and the expectation is taken over the trajectory starting from time step t. k is a variable used to represent the time step within the infinite sum. The equation calculates the expected cumulative reward (discounted by γ) starting from time step t for an infinite number of time steps, given the state s_t at time step t. In other words, it calculates the expected return starting from the current state s_t until the end of the episode (Schulman et al., 2017). The PPO algorithm typically uses a trust region optimization method to update the policy parameters, such as Conjugate Gradient or Adam (Engstrom et al., 2020; Kingma and Ba, 2014). In Algorithm 1, we provide a high-level overview of the PPO approach.

Algorithm 1: PPO inspired by Schulman et al. (2017)

Input: Policy π_θ with parameters θ, value function V_ϕ with parameters ϕ
Output: Policy π_θ with updated parameters θ, value function V_ϕ with updated parameters ϕ
for *each iteration* **do**
  Collect a set of trajectories $D = \tau_1, \tau_2, \ldots, \tau_N$ using the current policy π_θ;
  Compute the advantages $A(\tau)$ for each trajectory $\tau \in D$;
  Update the value function parameters ϕ by minimizing the mean-squared error loss $\mathcal{L}\text{VF} = \frac{1}{|D|} \sum \tau \in D \sum_{t=0}^{T-1} (V_\phi(s_t) - G_t)^2$;
  for *each epoch* **do**
    Shuffle the trajectories in D;
    Divide D into minibatches of size m;
    for *each minibatch* **do**
      Compute the clipped surrogate objective function $L_{\text{clip}}(\theta)$ using the minibatch data;
      Update the policy parameters θ using a trust region optimization method that maximizes $L_{\text{clip}}(\theta)$;
    end
  end
end

In Algorithm 1, τ represents a trajectory, s_t represents the state at time step t, T represents the length of a trajectory, G_t represents the discounted sum of rewards from time step t to the end of the trajectory, m represents the minibatch size, and $L_{\text{clip}}(\theta)$ is the clipped surrogate objective function defined in Eq. (3).

4.2. Approach 2: Rolling-horizon stochastic programming batch-planning

An alternative approach for addressing the DSTBPP is to use a rolling-horizon batch-planning heuristic method that employs Two-Stage Stochastic MIP, referred to as SP throughout this paper. To implement this approach, we execute an SP model at the end of each day for all the containers that should be planned, based on known data at the end of day $d \in D$, as well as information on any delayed containers up to the present.

Table 3
List of variables and elements.

Sets	
Ω	set of generated scenarios, $\omega \in \{1, 2, \ldots, \lvert\Omega\rvert\}$
Second stage decision variables	
$R_i^{d,\omega} \in \{0,1\}$	container $i \in \hat{L}$ is removed from own truck starting on day $d \in D$
$A_i^{d,\omega} \in \{0,1\}$	container $i \in \hat{L}$ is re-planned to own truck starting on day $d \in D$
$B_i^{\omega} \in \{0,1\}$	container $i \in \hat{L}$ is re-planned to charter truck
Parameters	
$late_i^{\omega}$	number of days container i arrives late in scenario ω
p_{ω}	probability of observing scenario ω

Within the context of the MDP formulation of the DSTBPP, the SP model can be interpreted as determining an action just before the transition to the new day. It considers a set of scenarios of future container delays of containers announced at the end of day d, denoted by $\omega \in \Omega$. To keep the approach somewhat computationally tractable, we do not sample future container arrivals, as this results in extremely long computation times. The associated parameters and decision variables are given in Table 3. Note that for readability, we partly reuse some variables from the MDP formulation. Furthermore, we let use $\hat{L}$ for the set of to-be-planned containers, including any postponed containers $\mho$ and already planned containers in the future $\hbar$, which can still be rescheduled. An overview of the notation is given in Table 3. The SP model can then be formulated as follows:

$$\min \quad \sum_{i\in\hat{L}}\sum_{d\in D} C^{\text{OWN}} \cdot x_i^d + \sum_{i\in\hat{L}} C^{\text{CHARTER}} \cdot y_i + p_\omega \left[\sum_{\omega\in\Omega}\sum_{i\in\hat{L}}\sum_{d\in D} C^{\text{OWN}}(A_i^{t,\omega} - R_i^{d,\omega}) + \sum_{\omega\in\Omega}\sum_{i\in\hat{L}} C^{\text{CHARTER}} \cdot B_i^{\omega} \right] \tag{7}$$

$$\text{s.t.} \quad \sum_{d\in D} x_i^d + y_i = 1 \qquad \forall i \in \hat{L} \tag{8}$$

$$x_i^d \cdot (d - e_i) \geq 0 \qquad \forall d \in D, i \in \hat{L} \tag{9}$$

$$x_i^d \cdot (d - (l_i - del_i)) \leq 0 \qquad \forall d \in D, i \in \hat{L} \tag{10}$$

$$\sum_{i\in\hat{L}} \sum_{d'=d-2del_i+1}^{d+1} x_i^{d'} \leq cap_d \qquad \forall d \in D \tag{11}$$

$$\sum_{d\in D} R_i^{d,\omega} = \sum_{d\in D} A_i^{d,\omega} + B_i^{\omega} \qquad \forall i \in \hat{L}, \omega \in \Omega \tag{12}$$

$$A_i^{d,\omega} \cdot d \geq A_i^{d,\omega} \cdot (e_i + late_i^{\omega}) \qquad \forall d \in D, i \in \hat{L}, \omega \in \Omega \tag{13}$$

$$A_i^{d,\omega} \cdot (e_i + late_i^{\omega} + del_i) \leq A_i^{d,\omega} \cdot l_i \qquad \forall d \in D, i \in \hat{L}, \omega \in \Omega \tag{14}$$

$$B_i^{\omega} \cdot (e_i + late_i^{\omega} + del_i) \leq B_i^{\omega} \cdot l_i \qquad \forall i \in \hat{L}, \omega \in \Omega \tag{15}$$

$$\sum_{d'=d-2del_i+1}^{d+1} X_i^{d'} - \sum_{d'=d-2del_i+1}^{d+1} R_i^{d',\omega} + \sum_{d'=d-2del_i+1}^{d+1} A_i^{d',\omega} \leq cap_d \qquad \forall d \in D, i \in \hat{L}, \omega \in \Omega \tag{16}$$

$$R_i^{d,\omega} \leq X_i^d \qquad \forall d \in D, i \in \hat{L}, \omega \in \Omega \tag{17}$$

$$\sum_{d\in D} R_i^{d,\omega} \leq late_i^{\omega} \qquad \forall i \in \hat{L}, \omega \in \Omega \tag{18}$$

$$x_i^d \cdot (d - e_i - late_i^{\omega}) + late_i^{\omega} \cdot R_i^{d,\omega} \geq 0 \qquad \forall d \in D, i \in \hat{L}, \omega \in \Omega \tag{19}$$

$$x_i^d \in \{0,1\} \qquad \forall d \in D, i \in \hat{L} \tag{20}$$

$$y_i, z_i \in \{0,1\} \qquad \forall i \in \hat{L} \tag{21}$$

$$R_i^{d,\omega}, A_i^{d,\omega} \in \{0,1\} \qquad \forall d \in D, i \in \hat{L}, \omega \in \Omega \tag{22}$$

$$B_i^{\omega} \in \{0,1\} \qquad \forall i \in \hat{L}, \omega \in \Omega \tag{23}$$

The objective function (7) minimizes the total transportation cost of current containers. The first two terms represent the first-stage costs of own and charter trucks. The remaining elements are the recourse actions associated with the containers part of the second-stage component, which incorporates recourse actions of re-planning containers. This includes the negative costs of containers removed from trucks (i.e., own trucks or charter trucks), costs of containers added to trucks, and costs of containers added to trucks across all scenarios. Constraint (8) ensures that each container is assigned to a truck. Furthermore, containers assigned to own trucks must be assigned on or after their earliest available day (Constraint (9)) and delivered on or before the latest arrival day (Constraint (10)). We model these domain restrictions in this way to prevent us from introducing additional notation. Additionally,

the number of trucks transporting or returning containers on a given day must not exceed the capacity (Constraint (11)). If a container is removed from an own truck, it must be added to an own or charter truck (Constraint (12)). Re-planned containers that were initially planned on an own truck must depart on or after their earliest availability day (note that the earliest leave day is delayed for these containers) (Constraint (13)). Re-planned containers that were initially planned on an own truck must arrive on or before their latest arrival day (Constraint (14)). Re-planned containers that were initially planned on a charter truck must arrive on or before their latest arrival day (Constraint (15)). Capacity constraints must also be met after re-planning (Constraint (16)). A container can only be removed from an own truck if it was initially planned on that truck (Constraint (17)). A container should only be re-planned if it is late (Constraint (18)). Finally, a container must be re-planned if it is too late for the truck on which it was initially planned (Constraint (19)). The remaining constraints indicate the variable domains.

5. Experiments and results

This section discusses the experiments we carried out to test our methods. We first introduce the experimental settings and the tested benchmark methods. Then, we discuss the training and stability of the PPO approach, and finally, we discuss the obtained results on solving the DSTBPP.

5.1. Experimental settings

Data. This experiment is designed on the basis of a practical case study at a logistics company. We generated data with properties that are based on the long-haul transportation planning problem from a single depot of the logistics company. These data have the following properties: the number of containers N that are announced each day are uniformly distributed between 2 and 5, the maximum truck capacity is 10, such that for each $d \in D$: $cap_d \leq 10$, and the temporal properties of containers are such that $e_i \in [1,7]$, $l_i \in [2,12]$, $del_i \in [1,5]$ are uniformly distributed. Charter trucks are always available and always reach the destination on time. Also, we assume a finite planning horizon of one month (i.e., 30 days), divided into equal-length time periods (i.e., days). We set the latest customer notification deadline k to one day before the expected container arrival. Finally, container delays are generated randomly based on two different levels of uncertainty 30% and 50%. These are the probabilities that container i has a one-day delay (i.e., $late_i = 1$). We also assume each scenario $\omega \in \Omega$ of the SP approach occurs with the same probability. We used the OpenAI Gym (Brockman et al., 2016) to simulate the dynamic and stochastic elements. Gym in Python is a framework that makes it easy to model problems according to the DRL cycle, as shown in Fig. 2, and subsequently provides the algorithms that can be used to solve them (especially PPO). Gym simplifies the creation of RL environments by offering a standardized API and a wide variety of pre-built environments, making it accessible even for those who may not be familiar with its intricacies. We introduced the required parameters and used the default parameters provided by Gym for the rest.

Training parameters. The neural network used for the PPO approach is trained using PPO with Masking. The training parameters are set with a gamma value of 1.0, number of steps of 4096, and a total of 10,000,000 episodes. The environment is monitored using Stable Baselines3 (SB3) monitor, and a check function with a frequency of 5000 is used to save the best model during training. It is important to note that the starting state of each episode is different from other episodes, and all operations must be planned in each episode, which is a full planning horizon. For validation, we tested the trained model on 20 full planning horizons, each consisting of 30 planning days and up to 100 containers as unseen instances. The remaining parameters are initialized according to their default values. The agent and the simulation model are executed on a machine with an Intel(R) Core(TM) i7 Processor CPU @ 2.80 GHz and 16 GB of RAM, with no graphics module used for training the neural network.

Tested methods. To benchmark the PPO approach, we employ the DQN method, rolling-horizon batch-planning SP approach, and a perfect information MIP method. Additionally, we evaluate various heuristics derived from the literature and based on insights gathered from discussions with our industry partner's current practices. The literature suggests that the commonly used constructive heuristics for the bin packing problem are the 'First-Fit' and 'Best-Fit' heuristics (Martello and Toth, 1990; Dyckhoff, 1990). This motivated us to consider three heuristics:

- 'First Fit' heuristic: This heuristic sorts containers based on their dispatching heuristics, then assigns containers to the first available own truck. If no trucks are available, a charter truck is used instead.
- 'Postpone - First Fit' heuristic: This heuristic also sorts containers based on their dispatching heuristics and postpones container assignments to the next planning day as much as possible within the constraints. It then assigns the containers to the first available own truck. If no trucks are available, a charter truck is used instead.
- 'Random': This heuristic also sorts containers based on their dispatching heuristics, then assigns containers randomly to one of the available trucks.

5.2. Training and stability analysis of PPO

During the learning process, we trained our PPO approach using both the FIFO and the EDF heuristic. To assess the effectiveness, we measured the total reward collected by the agent in each episode during training, following the approach suggested by Bellemare et al. (2013). It is worth noting that similar trends were observed in all other tested settings. Fig. 3 illustrates the changes in the total reward per episode, highlighting a consistent improvement in rewards throughout the training process. This indicates that our training did not encounter any divergence issues and successfully converged to higher rewards. Furthermore, Fig. 3 provides visual evidence that our DRL agent effectively learned container allocation patterns and progressively moved towards better solutions as it underwent training.

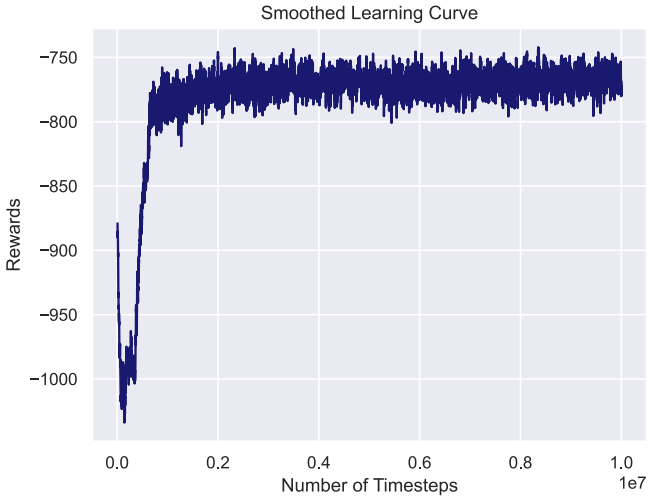


Fig. 3. Illustrative example of the training process.

5.3. Methods comparison and optimality evaluation

We conducted an extensive series of experiments to evaluate the effectiveness of diverse capacity planning methods in two distinct operational settings. The first setting adopts a fully real-time/online configuration, employing the FIFO dispatching heuristic, which necessitates no additional information about other containers and allows immediate planning as soon as an individual container is announced to the system. In contrast, the second setting involves end-of-the-day planning using the EDF heuristic, which requires information about the due dates of all containers. In this heuristic, containers are prioritized based on their due dates, and planning is exclusively carried out at the end of each day.

Our evaluation uses the performance indicators *transportation costs*, *own trucks capacity utilization*, and *computational time*. These indicators were measured under two distinct levels of uncertainty across 20 replications. The results of the experiments conducted in the FIFO-based setting are presented in Fig. 4(a), 4(b), 4(c), and 4(d). On the other hand, the results from the experiments in the EDF-based setting are illustrated in Fig. 5(a), 5(b), 5(c), and 5(d). We refer to the perfect information MIP approach as a 'benchmark'. Additionally, the general computational time is showcased in Fig. 6.

Figs. 4(a), 4(b) and 5(a), and 5(b) demonstrate that SP-based rolling-horizon batch-planning methods with different numbers of scenarios consistently outperform other methods and approach the benchmark perfect information solution in both levels of uncertainty. Despite planning one container at a time and having limited knowledge about other containers, the PPO algorithm achieves performance that is close to the SP-based rolling-horizon batch-planning methods and outperforms the simple heuristics in all instances. The 'Postpone-First Fit' heuristic performed better than the other heuristics, by postponing container assignments to the next planning day as much as possible, thereby reducing the risk of re-planning.

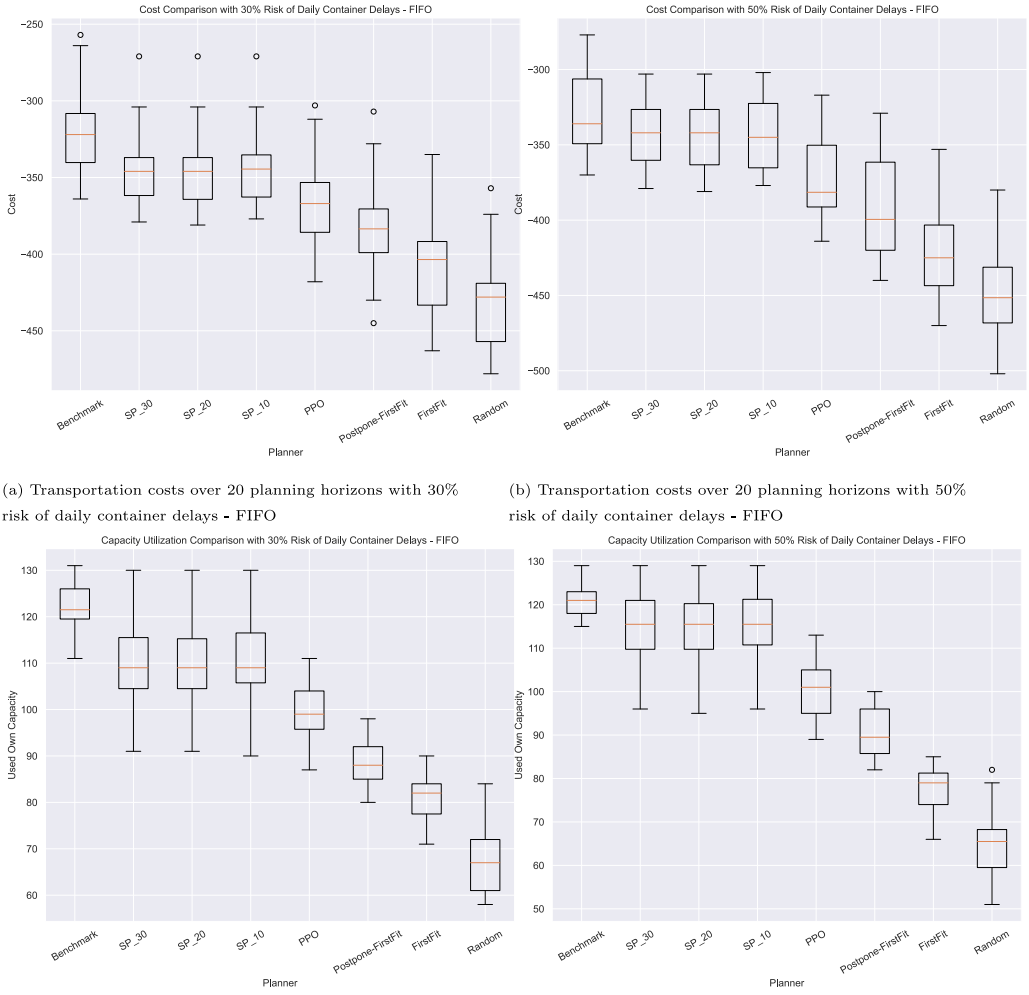
According to Fig. 4(c), 4(d), 5(c), and 5(d) the SP-based rolling-horizon batch-planning methods consistently utilized a greater amount of own truck capacity compared to other methods and were closer to the benchmark perfect information solution in both uncertainty levels. This finding is notable as these methods plan one container at a time and have limited knowledge about other containers. The PPO algorithm utilizes a similar amount of capacity to that of the SP-based rolling-horizon batch-planning methods. In contrast, the 'Postpone-First Fit' heuristic utilized more capacity than the other heuristics, such as 'Postpone-First Fit' and 'Random Assignment'. However, despite this strategy, there was still a significant quality gap with the benchmark solutions. It is worth noting that there is an inverse correlation between capacity utilization and total cost of transport, as the cost of transporting containers with own trucks is lower than with charter ones.

Fig. 6 shows the total computational time for each method over 20 replications with a varying number of orders per day. The heuristics made decisions relatively quickly, as did the PPO algorithm. As expected, the SP-based rolling-horizon batch-planning methods were much slower than the other methods, and an increase in the number of containers and scenarios would exacerbate these differences.

5.3.1. Real-time planning setting

We continue by looking closer to the real-time planning setting utilizing the FIFO dispatching rule. We focus on the methods' responsiveness to uncertainties and their capacity to mitigate the adverse impacts arising from delays. Table 4 reports differences in the average of the total costs and total computational time of each method in two levels of uncertainty over 20 replications in the real-time FIFO planning setting.

Under the 30% delay scenario, the Random method exhibits a 34% higher carrier cost compared to the benchmark solution with perfect information. This outcome is reasonable since the Random method lacks a specific allocation policy and only adheres to



(a) Transportation costs over 20 planning horizons with 30% risk of daily container delays - FIFO

(b) Transportation costs over 20 planning horizons with 50% risk of daily container delays - FIFO

(c) Own truck capacity utilization with 30% risk of daily container delays over 20 planning horizons - FIFO

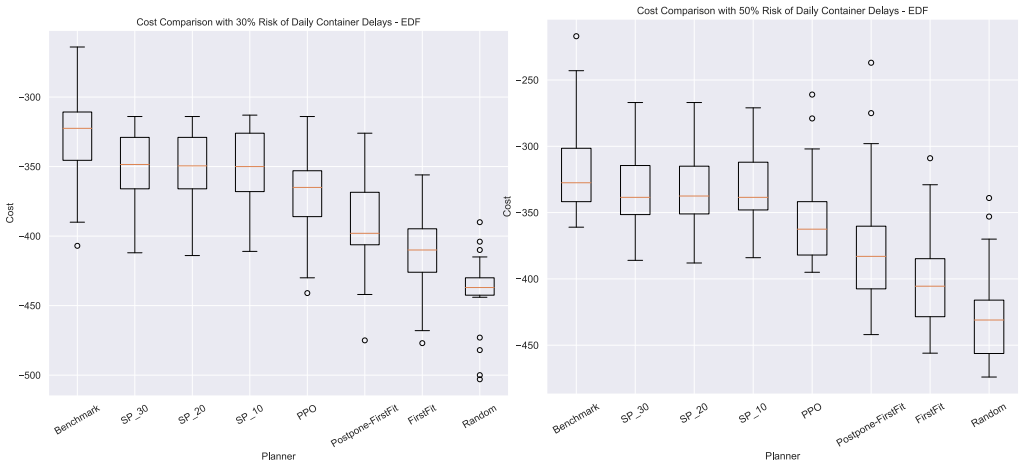
(d) Own truck capacity utilization with 50% risk of daily container delays over 20 planning horizons - FIFO

Fig. 4. Performance of methods under FIFO dispatching heuristic.

Table 4
Average results over test 20 replications — Online (FIFO Policy).

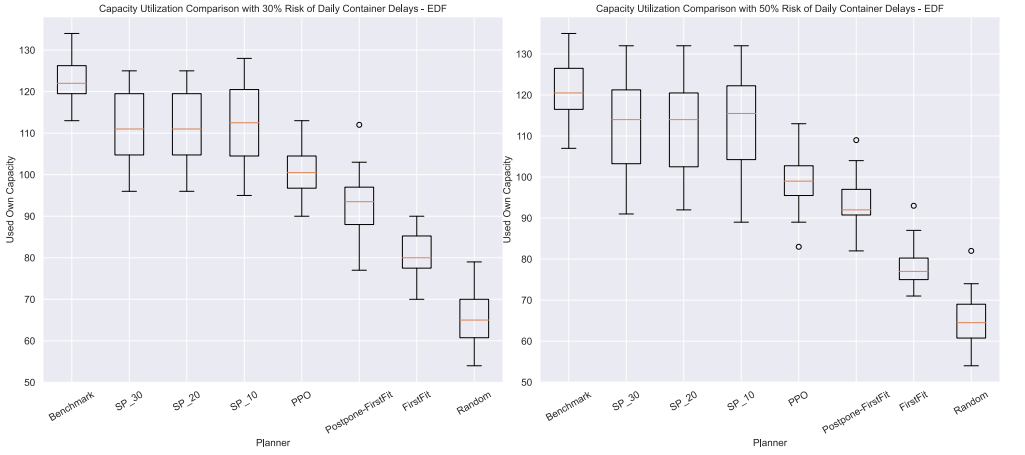
Method	30% delay per day			50% delay per day		
	Cost	Time	Capacity	Cost	Time	Capacity
Random	430	0.04	68	449	0.05	65
First Fit	406	0.03	81	403	0.16	82
Postpone-First Fit	384	0.06	88	392	0.06	90
PPO	366	0.32	99	371	0.34	100
SP-10 Scenarios	343	17.52	110	343	17.93	115
SP-20 Scenarios	345	34.73	110	344	35.94	115
SP-30 Scenarios	344	53.53	110	343	55.14	115
Benchmark	321	0.67	122	331	0.62	121

planning constraints. It shows a considerable -93% difference in processing time and a 47% underutilization of our own carrier's full truckload capacity, despite the benchmark solution's average utilization of 122 units of this capacity. Similarly, the First Fit method demonstrates a 26% cost differential, along with a -96% time disparity and a -33% capacity variance. This improvement over the Random approach stems from its greedy planning policy, which is more methodical than Random's ad hoc allocation. The Postpone-First Fit approach presents a 20% cost difference, a -91% time difference, and a -27% capacity difference. This approach enhances upon the First Fit method by strategically postponing allocations to align with more accurate arrival information, thereby



(a) Transportation costs over 20 planning horizons with 30% risk of daily container delays - EDF

(b) Transportation costs over 20 planning horizons with 50% risk of daily container delays - EDF



(c) Own truck capacity utilization with 30% risk of daily container delays over 20 planning horizons - EDF

(d) Own truck capacity utilization with 50% risk of daily container delays over 20 planning horizons - EDF

Fig. 5. Results for EDF Scheme.

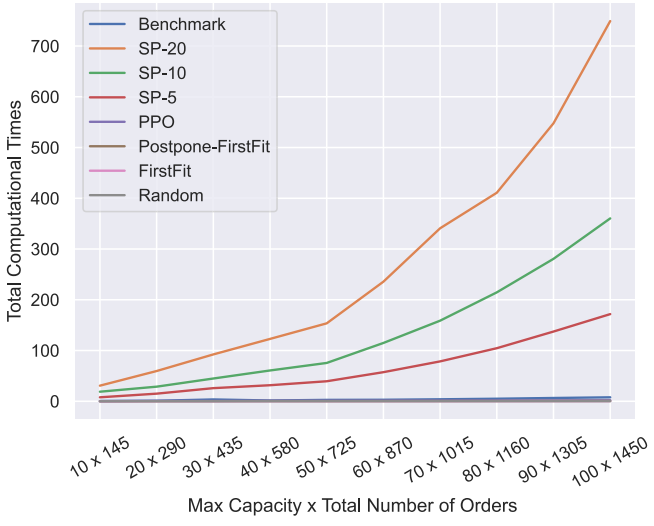


Fig. 6. Total computational times under different capacity and order quantities.

Table 5
Average results over test 20 replications — End of the day (EDF Policy).

Method	30% delay per day			50% delay per day		
	Cost	Time	Capacity	Cost	Time	Capacity
Random	441	0.05	66	428	0.04	65
First Fit	413	0.05	81	399	0.05	78
Postpone-First Fit	392	0.08	93	372	0.07	94
PPO	372	0.37	101	355	0.35	99
SP-10 Scenarios	351	20.35	112	332	17.36	113
SP-20 Scenarios	352	40.40	111	333	33.29	112
SP-30 Scenarios	352	61.08	111	332	50.88	113
Benchmark	329	0.71	123	315	0.63	121

reducing the risk of delays and re-planning. However, this strategy may not always be optimal, as it could delay carrier availability unnecessarily, hindering efficiency in some cases where carriers could be utilized sooner for other orders. In all cases, due to the use of simple heuristics, it is expected that computational time will be superior compared to mathematical optimization approaches. In stark contrast to these heuristics, the PPO method demonstrates a significantly narrower 14% cost differential, coupled with a -52% time disparity and a -18% capacity difference. The learned allocation policy by PPO proves much more efficient in terms of carrier cost and own capacity utilization. In Section 5.5, we delve into the PPO policy, comparing it with the Postpone-First Fit approach and discussing their distinctions. On the other hand, the SP planners, including SP-10 Scenarios, SP-20 Scenarios, and SP-30 Scenarios, show cost advantages averaging around 7%. However, their performance suffers from substantial increases in time differences ranging from 2515% to 7890%. This outcome is expected as SP approaches prioritize cost and capacity efficiency and are closer to baseline approaches due to having more comprehensive container information. Nevertheless, they face challenges in terms of computational time.

In the more rigorous 50% delay scenario, the PPO method maintains its performance benefits, registering a 12% cost difference, a -45% time difference, and a -17% capacity difference. By comparison, the SP planners continue to demonstrate their inherent cost advantages at around 4%, yet the pronounced escalation in time differences (2792% to 8794%) remain noteworthy.

5.3.2. End-of-the-day planning setting

Our evaluation extends seamlessly to the end-of-the-day planning setting, where the methods' performances are examined under both delay scenarios.

Table 5 reports differences in the average of the total costs and total computational time of each method in two levels of uncertainty over 20 planning horizons in the end-of-the-day planning setting.

In the 30% delay scenario, the Random method presents a 34% cost difference, a -93% time difference, and a -47% capacity difference of our own carrier's full truckload, despite the benchmark solution's average utilization of 123 units of the own trucks. Similarly, the First Fit method demonstrates a 26% cost difference, a substantial -94% time difference, and a 34% capacity difference. The Postpone-First Fit method showcases a 19% cost difference, an -89% time difference, and a more pronounced -24% capacity difference. In contrast, the PPO method has a 13% cost difference, a -48% time difference, and a -18% capacity difference. As anticipated, our PPO end-of-day planning shows improvement over the FIFO real-time approach in both scenarios compared to the benchmark. This is because we have visibility into the priority of containers for a specific planning day based on their due dates. This reduces the risk of missing due dates with our own carriers and the need for charter options, which requires due date information for all containers, a process that only completes at the end of the day. The SP approaches maintain their trends with cost advantages of around 7% coupled with amplified time differences (2766% to 8503%). Finally, in the 50% delay scenario, the PPO method showcases its resilience with a 13% cost difference, a -44% time difference, and a -18% capacity difference. The SP planners continue to maintain their cost advantages (5% to 6%), and high time differences (2656% to 7976%).

In summary, while the SP planners may initially appear attractive due to their potential cost advantages, they require more capacity than the PPO approach. Moreover, in many practical situations, as the one motivating our research, the reliance on MIP solvers that batch orders and thereby delay decision-making is not preferred. Interpreting the SP approach as a computational information relaxation bound on the PPO approach, we can infer that the PPO – given the information it can utilize to anticipate container delays – is an attractive approach for decision-makers in practice as it can provide high-quality solutions. Also, PPO can provide decisions in negligible computation, as opposed to the SP approaches.

5.4. Uncertainty level impact on real-time methods performance

We examined the impact of varying levels of uncertainty on the performance of real-time methods. As depicted in Fig. 7, we compared two PPO methods – one trained in an environment with 30% uncertainty and the other with 50% uncertainty – against the best-performing heuristic — the postponed-first fit approach. To illustrate the effects of different uncertainty levels, we progressively increased the test environment's uncertainty from 10% to 90%, conducting 500 replications at each level to ensure robust results. Fig. 7 reveals that the PPO method trained with 30% uncertainty is more adversely affected by increasing uncertainty levels compared to the PPO method trained with 50% uncertainty. Despite this, both PPO methods exhibit a significant performance

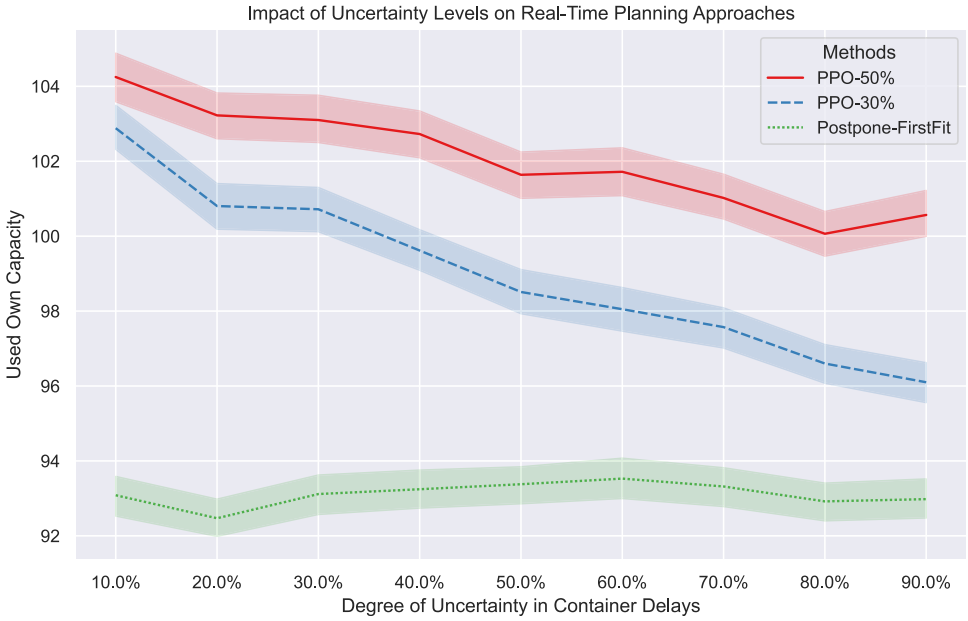


Fig. 7. Own truck capacity utilization under different risk levels of daily container delays over 500 planning horizons for each risk level.

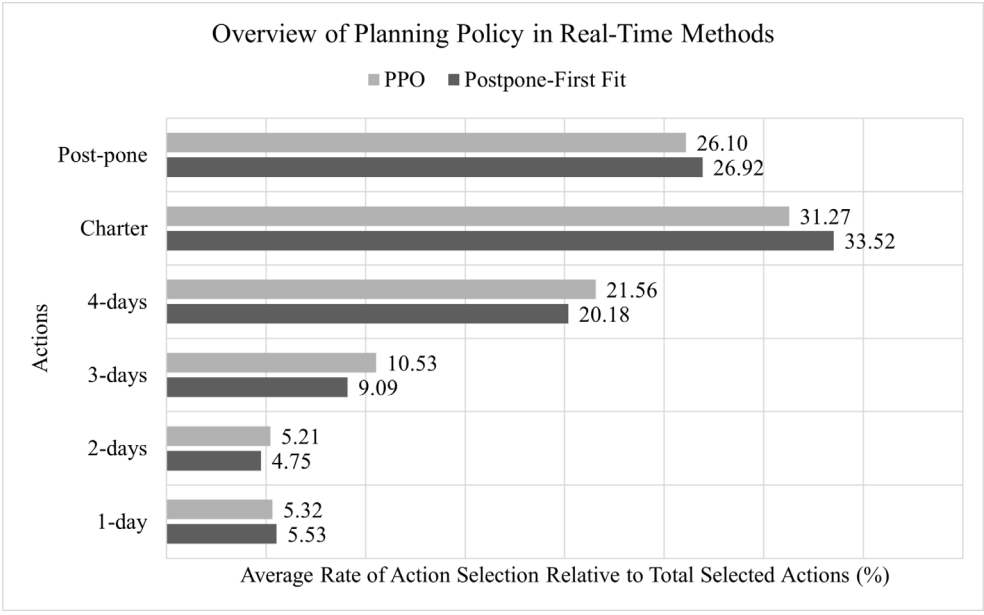
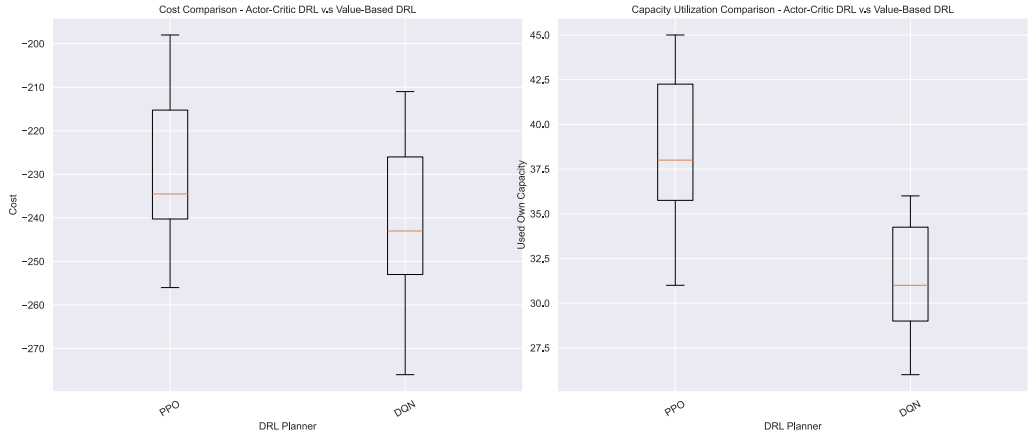


Fig. 8. Employed policy in real-time planning methods based on the average rate of action selection relative to total selected actions over 10 test replications.

gap in capacity usage when compared to the postponed-first fit heuristic. As uncertainty rises, the number of containers requiring re-planning increases, leading to higher transport costs and reduced utilization of own capacity. An interesting observation is that both PPO methods, especially the one trained using samples with 50% uncertainty, perform well even in environments with uncertainty levels they were not explicitly trained for. This suggests that our PPO method consistently outperforms heuristic approaches across all levels of uncertainty, demonstrating robust generalization to unforeseen samples and untrained scenarios.

5.5. Planning policy analysis in real-time methods

We analyze the planning policy learned through PPO and the planning strategies employed by the best-tested heuristic in our experimentation, the Postpone-First Fit technique. To this end, Fig. 8 depicts the average rate of action selection relative to total



(a) Transportation costs over 20 planning horizons with 30% risk of daily container delays

(b) Own truck capacity utilization over 20 planning horizons with 30% risk of daily container delays

Fig. 9. Performance comparison of the DRL methods.

number of selected actions across 10 tests. The possible actions are: selecting available trucks for departure within 1, 2, 3, or 4 days after the planning day, assigning an order to a charter truck, or postponing assignment to the next planning cycle. The figure shows that, on average, the policy learned via PPO is very similar to the Postpone-First Fit heuristic, but exhibits a 0.82% decrease in selecting the postponement option. Instead, PPO promptly allocates available trucks, leading to a 2.25% reduction in charter truck utilization. This proactive approach effectively aids in mitigating additional transportation costs stemming from cost differences between own trucks and charter trucks.

5.6. Comparing the performance of DRL algorithms

While we used the PPO algorithm to train the DRL agent, there are other algorithms as well. To illustrate the robustness of our conclusions for different DRL algorithms, we compare the performance of two prominent DRL algorithms: PPO (Schulman et al., 2017), an actor–critic method, and Deep Q-Network (DQN) (Mnih et al., 2015), a value-based approach. Their effectiveness in planning tasks is assessed using the metrics of cost and own resource utilization. As depicted in Fig. 9, PPO exhibits superior performance on both metrics, underscoring its potential as a more efficient solution for planning problems and supporting our selection of this DRL approach.

We clearly observe that PPO is superior to DQN, particularly because it offers better stability and performance in policy optimization. While DQN is a value-based method that estimates Q-values to guide policy updates, it often struggles with instability due to large, sudden updates that can lead to divergence (Li, 2023). PPO is an actor–critic method, combining both policy-based and value-based approaches. The actor component directly optimizes the policy by determining the best actions to take in each state, while the critic component estimates the value function to evaluate how good the current policy is. PPO addresses instability by introducing a clipping function that restricts the policy updates to a specific range, ensuring that the agent does not deviate too far from its previous behavior (Schulman et al., 2017). This clipping mechanism helps prevent large, destabilizing policy shifts, allowing for more gradual and stable policy improvements. Additionally, PPO employs a surrogate objective function, which is a conservative approximation of the policy gradient. This ensures that the optimization process is more reliable and efficient compared to DQN's approach, which relies on direct Q-value estimation. These factors collectively contribute to PPO's superior performance and stability across a wide range of reinforcement learning tasks.

6. Conclusion

In this paper, we studied the capacity planning problem within the dynamic and uncertain environment of a corridor-based integrated logistics system. We introduced the DSTBPP, which is the canonical optimization problem underlying the planning in such corridor-based systems. It concerns the real-time assignment of stochastically arriving containers to a fleet of own and outsourced trucks while adhering to stochastic deadline constraints. Our main contribution lies in the formulation of the problem as a MDP, and the subsequent proposal of a suite of solution methods. We developed a novel real-time planning algorithm based on PPO, a DRL technique. This method was complemented by a SP method in a rolling horizon batch planning setting. In addition, we presented benchmark solutions by comparing the performance of these algorithms with heuristics, and a computational regret bound in the form of an algorithm that has perfect information (i.e. assumes all delays are known in advance) and uses a MIP method to plan the transport orders with that information.

The evaluation of our proposed methods encompassed scenarios with 30% and 50% delays, both in real-time and in end-of-the-day batch planning settings. Our analysis revolved around the effectiveness and efficiency of these methods as measured by their transportation cost and computational time complexity.

The results showcased the efficacy of our PPO algorithm. Additionally, our comparison with a state-of-the-art DQN approach further confirms that PPO is an excellent choice for a DRL-based algorithm. This method not only outperforms practical heuristics but also scales better to larger problems compared to the SP models for rolling-horizon batch planning. The PPO approach yields an average cost reduction of around 5% and increased utilization of own truck capacity in the range of 5% to 11%, when contrasted with practical heuristics. The PPO approach produces transportation costs that are around 6% worse than the best SP method, but does so at a computational performance that is two orders of magnitude better in our experimental setting, further increasing to a three-orders of magnitude improvement as the problem size is increased to a practical scale.

We hypothesize that the performance of the PPO algorithm, compared to heuristics, comes from its ability to learn patterns of potential delay, which it can use to proactively optimize resource allocation positions. Clearly the computational performance, compared to SP approaches, comes from the fact that SP approaches rely on evaluating multiple scenarios, which greatly increases computational complexity, while the PPO approach – once trained – does not have to do that. It is evident that the PPO offers a practical edge over the SP planners' reliance on predefined scenarios by providing a good balance between cost reduction and time complexity.

Consequently, the applicability of PPO across diverse contexts and scenarios warrants further exploration. Integrating machine learning and data-driven techniques could enhance its predictive capabilities, thus improving its performance. Moreover, the ability of PPO to plan by-the-order rather than in-batch also means that it has the potential to support transportation planners in dynamically adapting a plan to unexpected events (i.e., arrival delays). This also deserves further investigation.

CRediT authorship contribution statement

Amirreza Farahani: Writing – review & editing, Writing – original draft, Visualization, Validation, Methodology, Formal analysis, Data curation, Conceptualization. **Laura Genga:** Writing – review & editing, Supervision. **Albert H. Schrotenboer:** Writing – review & editing, Supervision, Funding acquisition, Conceptualization. **Remco Dijkman:** Writing – review & editing, Supervision, Resources, Methodology, Funding acquisition, Conceptualization.

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

The work leading to this paper is supported by the Connecting Europe Facility of the European Union under The Innovation and Networks Executive Agency (INEA), through the FENIX Project (grant number INEA/CEF/TRAN/M2018/1793401). Albert H. Schrotenboer has received support from the Dutch Science Foundation (NWO) through grant VI.Veni.211E.043.

References

Alves, C., De Carvalho, J.V., 2007. Accelerating column generation for variable sized bin-packing problems. European J. Oper. Res. 183 (3), 1333–1352. http://dx.doi.org/10.1016/j.ejor.2005.07.033.

Bai, R., Chen, X., Chen, Z.L., Cui, T., Gong, S., He, W., Jiang, X., Jin, H., Jin, J., Kendall, G., et al., 2023. Analytics and machine learning in vehicle routing research. Int. J. Prod. Res. 61 (1), 4–30. http://dx.doi.org/10.1080/00207543.2021.2013566.

Baldi, M.M., Crainic, T.G., Perboli, G., Tadei, R., 2012. The generalized bin packing problem. Transp. Res. E 48 (6), 1205–1220. http://dx.doi.org/10.1016/j.tre.2012.06.005.

Baldi, M.M., Crainic, T.G., Perboli, G., Tadei, R., 2014. Branch-and-price and beam search algorithms for the variable cost and size bin packing problem with optional items. Ann. Oper. Res. 222, 125–141. http://dx.doi.org/10.1007/s10479-012-1283-2.

Baldi, M.M., Manerba, D., Perboli, G., Tadei, R., 2019. A generalized bin packing problem for parcel delivery in last-mile logistics. European J. Oper. Res. 274 (3), 990–999. http://dx.doi.org/10.1016/j.ejor.2018.10.056.

Beeks, M., Afshar, R.R., Zhang, Y., Dijkman, R., Van Dorst, C., De Looijer, S., 2022. Deep reinforcement learning for a multi-objective online order batching problem. In: Proceedings of the International Conference on Automated Planning and Scheduling, Vol. 32. pp. 435–443. http://dx.doi.org/10.1609/icaps.v32i1.19829.

Bellemare, M.G., Naddaf, Y., Veness, J., Bowling, M., 2013. The arcade learning environment: An evaluation platform for general agents. J. Artificial Intelligence Res. 47, 253–279. http://dx.doi.org/10.1613/jair.3912.

Bianchi, L., Dorigo, M., Gambardella, L.M., Gutjahr, W.J., 2009. A survey on metaheuristics for stochastic combinatorial optimization. Nat. Comput. 8, 239–287. http://dx.doi.org/10.1007/s11047-008-9098-4.

Bikker, I.A., Mes, M.R., Sauré, A., Boucherie, R.J., 2020. Online capacity planning for rehabilitation treatments: an approximate dynamic programming approach. Probab. Engrg. Inform. Sci. 34 (3), 381–405. http://dx.doi.org/10.1017/S0269964818000402.

Blum, C., Roli, A., 2003. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. ACM Comput. Surv. (CSUR) 35 (3), 268–308. http://dx.doi.org/10.1145/937503.937505.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., Zaremba, W., 2016. Openai gym. http://dx.doi.org/10.48550/arXiv.1606.01540, arXiv preprint arXiv:1606.01540.

Casazza, M., Ceselli, A., 2016. Exactly solving packing problems with fragmentation. Comput. Oper. Res. 75, 202–213. http://dx.doi.org/10.1016/j.cor.2016.06.007.

Correia, I., Gouveia, L., Saldanha-da Gama, F., 2008. Solving the variable size bin packing problem with discretized formulations. Comput. Oper. Res. 35 (6), 2103–2113. http://dx.doi.org/10.1016/j.cor.2006.10.014.
Crainic, T.G., Fomeni, F.D., Rei, W., 2021. Multi-period bin packing model and effective constructive heuristics for corridor-based logistics capacity planning. Comput. Oper. Res. 132, 105308. http://dx.doi.org/10.1016/j.cor.2021.105308.
Crainic, T.G., Gobbato, L., Perboli, G., Rei, W., 2016. Logistics capacity planning: A stochastic bin packing formulation and a progressive hedging meta-heuristic. European J. Oper. Res. 253 (2), 404–417. http://dx.doi.org/10.1016/j.ejor.2016.02.040.
Crainic, T.G., Gobbato, L., Perboli, G., Rei, W., Watson, J.P., Woodruff, D.L., 2014. Bin packing problems with uncertainty on item characteristics: An application to capacity planning in logistics. Procedia-Soc. Behav. Sci. 111, 654–662. http://dx.doi.org/10.1016/j.sbspro.2014.01.099.
Crainic, T.G., Perboli, G., Rei, W., Tadei, R., 2011. Efficient lower bounds and heuristics for the variable cost and size bin packing problem. Comput. Oper. Res. 38 (11), 1474–1482. http://dx.doi.org/10.1016/j.cor.2011.01.001.
Dall'Orto, L.C., Crainic, T.G., Leal, J.E., Powell, W.B., 2006. The single-node dynamic service scheduling and dispatching problem. European J. Oper. Res. 170 (1), 1–23. http://dx.doi.org/10.1016/j.ejor.2004.06.016.
Dell'Amico, M., Furini, F., Iori, M., 2020. A branch-and-price algorithm for the temporal bin packing problem. Comput. Oper. Res. 114, 104825. http://dx.doi.org/10.1016/j.cor.2019.104825.
Dolan, E.D., Moré, J.J., 2002. Benchmarking optimization software with performance profiles. Math. Program. 91, 201–213. http://dx.doi.org/10.1007/s101070100263.
Durbin, M., Hoffman, K., 2008. OR PRACTICE—The dance of the thirty-ton trucks: Dispatching and scheduling in a dynamic environment. Oper. Res. 56 (1), 3–19. http://dx.doi.org/10.1287/opre.1070.0459.
Dyckhoff, H., 1990. A typology of cutting and packing problems. European J. Oper. Res. 44 (2), 145–159. http://dx.doi.org/10.1016/0377-2217(90)90350-K.
Engstrom, L., Ilyas, A., Santurkar, S., Tsipras, D., Janoos, F., Rudolph, L., Madry, A., 2020. Implementation matters in deep policy gradients: A case study on ppo and trpo. http://dx.doi.org/10.48550/arXiv.2005.12729, arXiv preprint arXiv:2005.12729.
Farahani, A., Van Elzakker, M., Genga, L., Troubil, P., Dijkman, R., 2023. Relational graph attention-based deep reinforcement learning: An application to flexible job shop scheduling with sequence-dependent setup times. In: International Conference on Learning and Intelligent Optimization. Springer, pp. 347–362. http://dx.doi.org/10.1007/978-3-031-44505-7_24.
Gao, K., Cao, Z., Zhang, L., Chen, Z., Han, Y., Pan, Q., 2019. A review on swarm intelligence and evolutionary algorithms for solving flexible job shop scheduling problems. IEEE/CAA J. Autom. Sin. 6 (4), 904–916. http://dx.doi.org/10.1109/JAS.2019.1911540.
Gumuskaya, V., van Jaarsveld, W., Dijkman, R., Grefen, P., Veenstra, A., 2021. Integrating stochastic programs and decision trees in capacitated barge planning with uncertain container arrivals. Transp. Res. C 132, 103383. http://dx.doi.org/10.1016/j.trc.2021.103383.
Heess, N., Tb, D., Sriram, S., Lemmon, J., Merel, J., Wayne, G., Tassa, Y., Erez, T., Wang, Z., Eslami, S., et al., 2017. Emergence of locomotion behaviours in rich environments. http://dx.doi.org/10.48550/arXiv.1707.02286, arXiv preprint arXiv:1707.02286.
Hildebrandt, F.D., Thomas, B.W., Ulmer, M.W., 2023. Opportunities for reinforcement learning in stochastic dynamic vehicle routing. Comput. Oper. Res. 150, 106071. http://dx.doi.org/10.1016/j.cor.2022.106071.
Kang, J., Park, S., 2003. Algorithms for the variable sized bin packing problem. European J. Oper. Res. 147 (2), 365–372. http://dx.doi.org/10.1016/S0377-2217(02)00247-3.
Kingma, D.P., Ba, J., 2014. Adam: A method for stochastic optimization. http://dx.doi.org/10.48550/arXiv.1412.6980, arXiv preprint arXiv:1412.6980.
Li, S.E., 2023. Deep reinforcement learning. In: Reinforcement Learning for Sequential Decision and Optimal Control. Springer, pp. 365–402. http://dx.doi.org/10.1007/978-981-19-7784-8_10.
Martello, S., Toth, P., 1990. Bin-packing problem. In: Knapsack Problems: Algorithms and Computer Implementations. John Wiley & Sons, Inc. USA, pp. 221–245.
Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., et al., 2015. Human-level control through deep reinforcement learning. Nature 518 (7540), 529–533. http://dx.doi.org/10.1038/nature14236.
Monaci, M., 2003. Algorithms for packing and scheduling problems. Q. J. Belg. Fr. Italian Oper. Res. Societies 1 (1), 85–87. http://dx.doi.org/10.1007/s10288-002-0011-1.
Perboli, G., Crainic, T.G., Lerma, V., 2018. Stochastic Bin Packing Models for Capacity Planning in Logistic Applications. CIRRELT, Centre interuniversitaire de recherche sur les réseaux d'entreprise, URL: https://www.cirrelt.ca/documentstravail/cirrelt-2018-25.pdf.
Perboli, G., Gobbato, L., Perfetti, F., 2014. Packing problems in transportation and supply chain: new problems and trends. Procedia-Soc. Behav. Sci. 111, 672–681. http://dx.doi.org/10.1016/j.sbspro.2014.01.101.
Perboli, G., Tadei, R., Baldi, M.M., 2012. The stochastic generalized bin packing problem. Discrete Appl. Math. 160 (7–8), 1291–1297. http://dx.doi.org/10.1016/j.dam.2011.10.037.
Pisinger, D., Sigurd, M., 2005. The two-dimensional bin packing problem with variable bin sizes and costs. Discrete Optim. 2 (2), 154–167. http://dx.doi.org/10.1016/j.disopt.2005.01.002.
Powell, W.B., 2019. A unified framework for stochastic optimization. European J. Oper. Res. 275 (3), 795–821. http://dx.doi.org/10.1016/j.ejor.2018.07.014.
Powell, W.B., 2021. From reinforcement learning to optimal control: A unified framework for sequential decisions. In: Handbook of Reinforcement Learning and Control. Springer, pp. 29–74. http://dx.doi.org/10.1007/978-3-030-60990-0_3.
Rivera, A.E.P., Mes, M.R., 2022. Anticipatory scheduling of synchromodal transport using approximate dynamic programming. Ann. Oper. Res. 1–35. http://dx.doi.org/10.1007/s10479-022-04668-6.
Schrotenboer, A.H., Buijs, P., van der Heide, G., Phoa, T., Kilic, O.A., 2020. A share-first-plan-second policy for collaboration in transportation networks. http://dx.doi.org/10.2139/ssrn.3601864, Available at SSRN 3601864.
Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O., 2017. Proximal policy optimization algorithms. http://dx.doi.org/10.48550/arXiv.1707.06347, arXiv preprint arXiv:1707.06347.
Sutton, R.S., Barto, A.G., 2018. Reinforcement Learning: An Introduction. MIT Press.
Tadumadze, G., Emde, S., 2022. Loading and scheduling outbound trucks at a dispatch warehouse. IISE Trans. 54 (8), 770–784. http://dx.doi.org/10.1080/24725854.2021.1983923.
Tseng, C.Y., Li, J., Lin, L.H., Wang, K., White, III, C.C., Wang, B., 2023. Deep reinforcement learning approach for dynamic capacity planning in decentralised regenerative medicine supply chains. Int. J. Prod. Res. 1–16. http://dx.doi.org/10.1080/00207543.2023.2262043.
Wäscher, G., Haußner, H., Schumann, H., 2007. An improved typology of cutting and packing problems. European J. Oper. Res. 183 (3), 1109–1130. http://dx.doi.org/10.1016/j.ejor.2005.12.047.
Zolfagharinia, H., Haughton, M., 2016. Effective truckload dispatch decision methods with incomplete advance load information. European J. Oper. Res. 252 (1), 103–121. http://dx.doi.org/10.1016/j.ejor.2016.01.006.