Research School for Operations Management and Logistics

Anticipatory Scheduling of Freight in a Synchromodal Transportation Network

A.E. Pérez Rivera, M.R.K. Mes

Beta Working Paper series 533

BETA publicatie	WP 533 (working paper)
ISBN	
ISSN	
NUR	
Eindhoven	October 2017

Anticipatory Scheduling of Freight in a Synchromodal Transportation Network

Arturo E. Pérez Rivera Martijn R.K. Mes

October 16, 2017

Abstract

We study the problem of scheduling freight transportation in a synchromodal network considering stochastic freight arrivals. In a synchromodal network, freights can be transported using any mode and any route as long as they arrive to their destination within their time-window. Furthermore, transportation plans can change at any decision moment given the actual circumstances. Performance is measured over the entire network and over time. We model this problem as a Markov Decision Process and propose a heuristic solution based on Approximate Dynamic Programming (ADP). Due to the multi-period nature of the problem, the one-step look-ahead perspective of traditional ADP designs can make the heuristic flounder and end in a local-optimum. To tackle this, we study the inclusion of Bayesian exploration using the Value of Perfect Information (VPI). In a series of numerical experiments, we show how VPI significantly improves a traditional ADP algorithm. Furthermore, we show how our proposed ADP-VPI combination achieves more than 20% gains over the profit achieved by common practice heuristics. Finally, we discuss our experience with merging VPI into ADP and elaborate on further research directions in the anticipatory scheduling of freight in a synchromodal transportation network.

Keywords: synchromodal transportation, anticipatory scheduling, approximate dynamic programming.

1 Introduction

In recent years, the interest of Logistic Service Providers (LSPs) in intermodal freight transportation has increased due to its potential savings in cost [16] and emissions [6] when compared to road transportation. However, economical and environmental benefits are not alone the precursors of a change from road to intermodal transportation. The organization and synchronization of the various transportation services in an intermodal network can further ease or impede this change. Increasingly, LSPs are opting for cooperation and integration approaches among different processes for planning and controlling the transportation of freight in an intermodal network [35]. In this paper, we focus on one of those approaches: synchromodal transportation.

Synchromodal transportation is intermodal transportation where the LSPs employ the available services synchronously, to bring freights to their destination

at agreed conditions such as time, costs, emissions, etc. [30]. In synchromodal transportation, any *service* (transportation mode with specific attributes such as schedule, duration, capacity, cost, etc.) and any *transfer* (change from one service to another in a terminal) can be used for any freight [30]. This increased flexibility allows LSPs to select, or to change, the next part of a freight's route as late as possible, with the possibility of including the latest information about the transportation network. As a result, there are more consolidation options throughout the network and throughout time than in traditional intermodal transportation. However, identifying which of these consolidation opportunities are good is more challenging than in intermodal transportation.

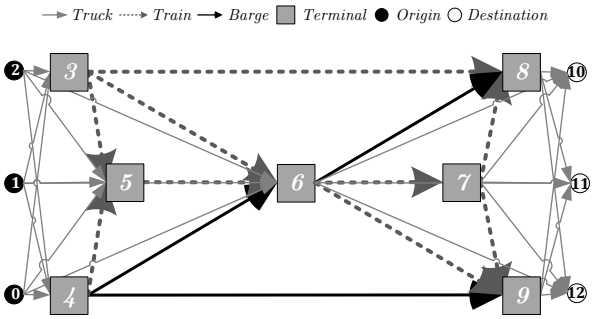

Figure 1: Example of an intermodal network with seven terminals

To exemplify the consolidation opportunities and their complexity in synchromodal transportation, consider a freight that has to be transported from 1 to 10 in Figure 1. This freight can be transported using 24 combinations of services across the seven terminals. In traditional intermodal transportation, freights usually have a pre-defined mode. If this would be the case for our freight, say barge only, then only 3 out of the 24 combinations would be feasible. Moreover, in intermodal transportation, freights usually require a fixed plan at the moment they are picked up at their origin. If this would be the case for our freight, say using the barge from 4 to 6 and subsequently from 6 to 8, a last-minute change at 6 would not be possible. This change may be desirable, for instance, if there is an unforeseen delay between 6 and 8, or when the train from 6 to 7 suddenly has cheaper space for transportation upon arrival at 6. Naturally, estimating whether a delay or cheaper space will occur at a future transfer in the network is challenging. However, looking ahead for these better consolidation opportunities can pay-off.

In this paper, we investigate scheduling methods that look ahead for consolidation opportunities in synchromodal transportation. The objective of our investigation is threefold: (i) to model the scheduling of freight in a synchromodal transportation network as a stochastic finite horizon optimization problem, (ii) to design a solution approach that handles the uncertainty and the time relations among parameters and variables in synchromodal scheduling, and (iii) to explore the use of our approach under various network configurations and freight demand patterns. The contribution of our work, following our objective,

is threefold. First, we model the scheduling problem using a Markov Decision Process (MDP) model, which maximizes the expected reward over time. Second, we design a heuristic solution algorithm for the MDP model using the framework of Approximate Dynamic Programming (ADP) and incorporating Bayesian learning, using the concept Value of Perfect Information (VPI), to cope with the exploration versus exploitation tradeoff in ADP. Our algorithm iteratively estimates the impact of the scheduling options in immediate and posterior performance, and constructs a solution policy that anticipates on the stochastic freight demand and the network-wise performance over time. Third, we characterize how traditional ADP designs can make the algorithm flounder and end in a local-optimum and we analyze how VPI elements can help the algorithm to overcome those problems. More specifically, we describe, test, and show the benefits of various modifications to the VPI concept in ADP.

The remainder of this paper is organized as follows. We begin by describing our problem, its characteristics and its challenges in Section 2. In Section 3, we examine relevant literature about scheduling synchromodal transportation. We formulate an MDP model for our problem in Section 4. Subsequently, we present our approach to solve the model using ADP and VPI in Section 5. In Section 6, we test our approach under different network settings and discuss the results. We finalize in Section 7 with our main conclusions and insights for further research.

2 Problem Description

We study the problem of scheduling freight in a synchromodal transportation network with the objective of maximizing performance over a multi-period horizon. In this problem, there are no restrictions on the services or transfers used to bring a freight to its destination and there are no fixed plans for freights. However, the destination, release-day and due-day of freights are fixed and known upon arrival of the freight. The number of freights that will arrive each period of the horizon and their characteristics are uncertain, but there is probabilistic information about their arrival (which may vary between periods of the horizon). On the supply side, we consider that all available services and transfers are fixed, but not necessarily the same, for each period of the horizon. We consider that a single network-wise LSP decides over all services and transfers even if they are not its own. We consider that the duration, cost, and revenue of any service may be spread over multiple periods in the horizon.

In general, the complexity of scheduling freight in synchromodal transportation lies in the relation between the possible decisions, and their effect on the performance over time. At any given period of the horizon (which we refer to as *day* in the remainder of this paper), there are three possibilities for scheduling a freight, independent of where it is located. The scheduler can either (i) transport a freight to its destination today, (ii) transport it to an intermodal terminal using a service available today, or (iii) postpone its transportation to another day. Part of the impact of these scheduling options on the performance can be measured immediately (e.g., transportation costs, revenue, holding costs, CO_2 emissions, capacity utilization). However, another part of their impact occurs on a posterior moment. For example, transporting a freight to its destination today reduces the number of freights to be considered for consolidation in the

future; transporting a freight to an intermodal terminal defines the future services that can be used for transporting it; and postponing a freight may reduce its feasible transportation options due to its time-window, or may saturate the network. The future impact of each scheduling option is dependent on posterior decisions, as well as the freights that will arrive in the future and their characteristics (which are uncertain). It is therefore essential to estimate the impact of current scheduling decisions on the future performance, and to anticipate on it.

Before formulating a model and solution approach, we study the relevant literature about the characteristics and challenges of scheduling decisions in synchromodal transportation in the following section.

3 Literature Review

In this section, we review literature related to scheduling freight in synchromodal transportation. First, we perform a brief literature review specifically on scheduling problems in synchromodal transportation. Since this literature is scarce, we subsequently focus our review on scheduling problems arising in dynamic and flexible inter/multi-modal transportation. These problems can conceptually be seen as precursors for synchromodal transportation. We briefly review the problem characteristics and proposed solutions of relevant intermodal transportation studies, and identify their shortcomings with respect to synchromodality. For an in-depth review of the scheduling problems arising in intermodal transportation, we refer the reader to [5], [35], and [10]. Third, we focus on literature using Approximate Dynamic Programming (ADP), a suitable approach for solving large-scale transportation problems with stochastic freight arrivals, and inspect its necessary changes when applied to our problem. We finalize this section by stating our contribution to the literature related to scheduling synchromodal and intermodal transportation, as well as using ADP for transportation problems.

Scheduling problems in synchromodality deal with flexibility in mode choice and with decisions based on real-time information [35]. These characteristics require that there is coordination between multiple network actors and an overview of transportation supply and freight demand in the network for its scheduling [10], and that a balance between demand and supply is made every time new information becomes known [30]. Most studies specifically about synchromodal scheduling, focus on the flexibility aspect while somewhat ignoring the real-time information aspect. For example, Behdani et al. [2] and Riessen et al. [31] determine the schedules of modes and the assignment of containers to the various modes assuming a deterministic demand without incorporating the effect of real-time information and previous decisions in their models. In other words, they take a reactive approach and assume that re-planning can be done by solving the model again once the new information becomes known. Studies that explicitly consider re-planning (e.g., due to new demand or disruptions), such as Zhang and Pel [45] and [22], have taken a more proactive approach towards re-planning but not explicitly anticipate on future real-time information. However, the studies that consider re-planning in intermodal transportation have shown that significant gains can be achieved with more proactive approaches.

Scheduling problems in intermodal transportation that are closely related

to synchromodal transportation can be categorized into two groups: (i) those that include dynamic and flexible assignment of freights to services and (ii) those that include anticipatory decisions on information that becomes known over time. A combination of these two constitutes scheduling synchromodal transportation. In the first group, problems fall under the intermodal scheduling family of Dynamic Service Network Design (DSND). In DSND methods, freights are assigned to transportation services and modes in a network where least one feature varies over time [35]. Graph theory and (integer) linear programming methods are commonly used to model DSND problems due to their time-space nature. However, these methods have limitations for large and complex time-evolving problems [43], which are common to synchromodality [30]. To tackle these limitations, decomposition algorithms [11], receding horizons [17], and model predictive control [23], have been combined with DSND models in the literature. One disadvantage of combining these constructs with DSND models is that the relation between new stochastic information and the decisions is harder to include. This may explain why the majority of DSND studies considers deterministic demand only [35] although the need to incorporate uncertainty in demand has been recognized [18].

In the second group, most studies have been about extending DSND models with stochastic demand to anticipate on new information that becomes known over time. For example, [18] and [7] have used scenario generation methods to create schedules that are robust to the various demand realizations (i.e., new freights arriving). However, the resulting schedule does not adapt to new information *dynamically* as other methods do. Adapting dynamically means new schedules depend on the information that became know. Methods such as two-stage stochastic programming [19, 1] and ADP [8, 24], which include re-planning with the new information that became known, have been shown to have benefits over static decisions. However, when considering synchromodal scheduling, some limitations arise. In two-stage stochastic programming, explicit probabilistic formulations and computational complexity limit the size of problem instances that can be solved. In ADP, the design and validation of the approximation algorithm is problem specific. Nevertheless, ADP reduces the computational complexity while providing a close-to-optimal solution and has been shown to work well in the scheduling of freights in intermodal transportation [25, 26] and single-mode transportation problems [34, 12, 39, 40]. Although ADP seems to be an ideal candidate to fill the gap in the literature about anticipatory scheduling of freight in synchromodal transportation, our problem requires more than its mere application.

In transportation problems such as ours, the complex, time-revealing, and stochastic nature of the network makes the application of ADP difficult [29]. To begin with, the multi-period traveling times are a known issue to traditional ADP algorithms in transportation problems [13]. Furthermore, when transportation services have multiple attributes (in our case, different capacities, durations, costs, revenues, beginning and ending locations), the design of the Value Function Approximation (VFA) and its learning-phase play a crucial role in ADP [34]. In the design of the VFA, additional methods to the common post-decision state in ADP, such as aggregation of post-decision attributes [4] and sampling [14], may be necessary. In the learning phase of the VFA, a fundamental challenge that arises is the exploration versus exploitation problem [27]. The problem consists on whether to let ADP make the best decision according

to its current estimate of the VFA (exploit) or let it make a different decision that may lead to an improvement of the VFA (explore).

The exploration versus exploitation dilemma has been widely studied in the reinforcement learning [37] and optimal learning communities [28]. The dilemma arises when a machine/agent tries to maximize its rewards by interacting with its environment through a series of actions. A widely studied optimization problem facing this dilemma is the so-called multi-armed bandit problem [21, 42]. Two approaches that have been applied to this problem, and the exploration versus exploitation dilemma, are evolutionary algorithms [41] and Bayesian learning methods [21, 36]. Bayesian learning methods usually rely on the concept of value of information [9], which appears under different names, among which expected improvement and knowledge gradient [33]. Although many real-life problems can be modeled as mutli-armed bandit problems or solved using evolutionary algorithms and Bayesian learning methods, there are several difficulties to translate these approaches to a transportation problem and to an ADP method as a solution approach. Among those difficulties we find the so-called "physical" state where decisions depend on the state of our physical resources, such as containers and barges, and the correlation of values of alternative decisions (e.g., economies of scale in adding more containers to a barge) [33]. These difficulties are incorporated by [33] assuming an infinite horizon, whereas our problem deals with a finite horizon. To the best of our knowledge, the application of Bayesian learning techniques within finite horizon ADP has not been studied before. Nevertheless, ADP can benefit from a translation of the knowledge on Bayesian learning to deal with the exploration versus exploitation dilemma [27, 33].

Overall, we observed in the literature that DSND models and methods provide a useful base for synchromodal scheduling with some additional work. We believe that our contribution to DSND methods and synchromodal scheduling has three focus points. First, we design an MDP model and solution method based on ADP that incorporates stochastic freight demand characteristics and complex time and performance evolution of the transportation network. Second, we explore the use of new exploration strategies for ADP based on Bayesian exploration, and provide design and validation insights. Third, we compare our best ADP design against a benchmark heuristic, under different problem characteristics, and specify further research directions based on the insights.

4 Markov Decision Process Model

In this section, we formulate the problem of scheduling freights in synchromodal transportation using a Markov Decision Process (MDP) model. An MDP model analyzes the performance over time as been partly under the control of the decision maker and partly random. We begin by introducing the notation and all required input parameters. Subsequently, we formulate the elements of the MDP model: stages, states, decisions, transitions, and objective function. Finally, we examine the relations between the various elements of our model and identify challenges that these relations bring for heuristic approaches to solve the model.

4.1 Notation

We consider a finite horizon $\mathcal{T}$ of $T^{\max}$ days, i.e., $\mathcal{T} = \{0, 1, 2, \ldots, T^{\max} - 1\}$. Although we refer to a period in the horizon as a "day", it is important to note that time can be discretized in any arbitrary interval as long as all time-dependent parameters are measured in that same interval. Each day $t \in \mathcal{T}$, the transportation network is represented by a directed graph $G_t = (\mathcal{N}_t, \mathcal{A}_t)$, as it is usually done in DSND models. Nodes $\mathcal{N}_t$ denote locations where services begin or end, and arcs $\mathcal{A}_t$ denote the services running from one location to another. To ease the formulation, we categorize nodes into three types: (i) origin nodes $\mathcal{N}_t^{\mathrm{O}}$, (ii) destination nodes $\mathcal{N}_t^{\mathrm{D}}$, and (iii) intermodal terminal nodes $\mathcal{N}_t^{\mathrm{I}}$, such that $\mathcal{N}_t = \mathcal{N}_t^{\mathrm{O}} \cup \mathcal{N}_t^{\mathrm{D}} \cup \mathcal{N}_t^{\mathrm{I}}$; and we index all nodes in $\mathcal{N}_t$ by i, j, and d. In this categorization, the sets of origin and destination nodes represent the possible starting and ending locations of freights, respectively, and are mutually exclusive with the set of intermodal terminals nodes. This separation of nodes applies to our model, but not necessarily to our problem. We further elaborate on this assumption, and how to overcome it, in Appendix A.

Each day t, new freights with different characteristics arrive to the network. Each freight that arrives has a known origin $i \in \mathcal{N}_t^{\mathrm{O}}$, destination $d \in \mathcal{N}_t^{\mathrm{D}}$, release-day $r \in \mathcal{R}_t$, and time-window $k \in \mathcal{K}_t$. The release-day of a freight counts the number of days in which a freight will be released after its arrival. The set $\mathcal{R}_t = \{0, 1, 2, \ldots, R_t^{\max}\}$ ranges from immediate release to $R_t^{\max}$ days before release. The time-window of a freight measures the number of days in which a freight must be at its destination *after* it has been released. The set $\mathcal{K}_t = \{0, 1, 2, \ldots, K_t^{\max}\}$ ranges from the same day a freight is released to $K_t^{\max}$ days after it is released. Although freights are unknown before they arrive, there is probabilistic knowledge about their arrival in each origin $i \in \mathcal{N}_t^{\mathrm{O}}$. In between two consecutive days $t-1$ and t, for origin $i \in \mathcal{N}_t^{\mathrm{O}}$, a total of $f \in \mathbb{N}$ freights arrive with probability $p_{f,i,t}^{\mathrm{F}}$. A freight that arrives between days $t-1$ and t in origin $i \in \mathcal{N}_t^{\mathrm{O}}$ has destination $d \in \mathcal{N}_t^{\mathrm{D}}$ with probability $p_{d,i,t}^{\mathrm{D}}$, release-day $r \in \mathcal{R}_t$ with probability $p_{r,i,t}^{\mathrm{R}}$, and time-window $k \in \mathcal{K}_t$ with probability $p_{k,i,t}^{\mathrm{K}}$.

In a similar fashion to the categorization of nodes, we categorize arcs into three types: (i) arcs between an origin and an intermodal terminal node $\mathcal{A}_t^{\mathrm{O}} = \big\{(i,j) | i \in \mathcal{N}_t^{\mathrm{O}} \text{ and } j \in \mathcal{N}_t^{\mathrm{I}}\big\}$, (ii) arcs between two intermodal terminal nodes $\mathcal{A}_t^{\mathrm{I}} = \big\{(i,j) | i, j \in \mathcal{N}_t^{\mathrm{I}}\big\}$, and (iii) arcs between an origin or an intermodal node, and a destination $\mathcal{A}_t^{\mathrm{D}} = \big\{(i,d) | i \in \mathcal{N}_t^{\mathrm{O}} \cup \mathcal{N}_t^{\mathrm{I}} \text{ and } d \in \mathcal{N}_t^{\mathrm{D}}\big\}$. For services beginning at an origin or ending at a destination (i.e., $\mathcal{A}_t^{\mathrm{O}}$ and $\mathcal{A}_t^{\mathrm{D}}$), we assume there is unlimited capacity every day. In other words, we assume that the pre- and end-haulage operations of our synchromodal network are not restrictive. Typically, LSPs use trucks for their pre- and end-haulage operations and are able to hire additional trucks if required. However, for each service between two intermodal terminals (i.e., $\mathcal{A}_t^{\mathrm{I}}$), we consider there is a *maximum capacity* of $Q_{i,j,t}$ freights and a *transportation duration* of $L_{i,j,t}^{A}$ days. In addition to the duration and capacity of each service, we consider that for each location $i \in \mathcal{N}_t$, there is a transfer duration of $L_{i,t}^{N}$ days, which includes all handling operations, and we assume unlimited handling capacity. Consequently, the total time required for the service between two locations $M_{i,j,t} = L_{i,t}^{N} + L_{i,j,t}^{A} + L_{j,t}^{N}$. Finally, we consider that between any two nodes, there is at most one arc (i.e., one service between two locations) and that all service durations are at least one (i.e., $M_{i,j,t} \geq 1 \forall (i,j) \in \mathcal{A}_t$). To represent more services between two locations, nodes

and services can be modified as explained in Appendix A.

With respect to the objective, we define a generic reward function $R_t(\cdot)$ to capture the immediate reward (i.e., reward at day t) of transporting freight in the network. For each arc $(i,j) \in \mathcal{A}_t$, or service between i and j, we include three components in the reward function: (i) a revenue $A_{i,j,d,t}$ per freight with destination d, (ii) a setup cost $B_{i,j,t}$ independent of the number of freights using the service, and (iii) a variable cost $C_{i,j,d,t}$ per freight with destination d. These components can also have a value of zero to model different financial conditions such as receiving the entire revenue of a freight at the beginning of transportation (i.e., $A_{i,j,d,t} = 0 \forall i \notin \mathcal{N}_t^{\mathrm{O}}$) or constant cost for reserved space in a service (i.e., $B_{i,j,t} > 0$ and $C_{i,j,d,t} = 0$ for the reserved service (i,j)). Although various optimization criteria can be modeled using these three components, we note that the reward function $R_t(\cdot)$ is not limited to them and can other key performance indicators.

4.2 Formulation

The *stages* at which decisions are made in our MDP model correspond to the days in the horizon, i.e., $t \in \mathcal{T}$. The *state* of the system $S_t \in \mathcal{S}$ is modeled as the vector of all freights, and their characteristics, that are present at each node and each arc of the network (i.e., freights available at a location or traveling to a location) at stage t. In the state vector, we denote freights at location $i \in \mathcal{N}_t^{\mathrm{O}} \cup \mathcal{N}_t^{\mathrm{I}}$, that have destination $d \in \mathcal{N}_t^{\mathrm{D}}$, release-day $r \in \mathcal{R}'_t$, and time-window $k \in \mathcal{K}_t$ with the integer variable $F_{i,d,r,k,t}$. Hence, the state is given by (1).

$$S_t = [F_{i,d,r,k,t}]_{\forall i \in \mathcal{N}_t^{\mathrm{O}} \cup \mathcal{N}_t^{\mathrm{I}}, d \in \mathcal{N}_t^{\mathrm{D}}, r \in \mathcal{R}'_t, k \in \mathcal{K}_t} \tag{1}$$

Note that we use a different set $\mathcal{R}'_t$ for the release-day r to model freights that are being transported to location i using $F_{i,d,r,k,t}$ in a computationally efficient way. We define $\mathcal{R}'_t = \left\{0, 1, 2, \ldots, \max\left\{R_t^{\max}, \max_{(i,j)\in\mathcal{A}_t^{\mathrm{I}}} M_{i,j,t}\right\}\right\}$ and use a *virtual time-window* to model freights that are en route. For example, if a freight is being transported to location i using a service that departs on Monday and arrives at i on Thursday, then this freight will be modeled in the state of Tuesday as a freight that will be available at i in two days, i.e., a freight with release-day $r = 2$. Furthermore, if the freight has a deadline of Friday, its time-window on Tuesday will be $k = 1$, i.e., one day after it arrives or, in terms of our virtual time-windows, it is *virtually released*. So, on Tuesday, instead of modeling the freight that is already released and being transported with $r = 0$ and $k = 4$, we model it with $r = 2$ and $k = 1$. We further elaborate on the use of virtual time-windows to capture the evolution of the network later on in this section.

At every stage, the planner decides which of the released freights (i.e., $r = 0$) to transport using a given service and which ones to postpone. Remind that if the planner decides to transport a freight, only the part of a freight's route to the first destination is fixed, which may be an intermodal terminal or its final destination. We model the *decision* with the vector x_t consisting of all freights that will be transported at stage t, as shown in (2a). We denote the number of freights that will be transported from location i to location j (i.e., using service $(i,j) \in \mathcal{A}_t$), that have destination $d \in \mathcal{N}_t^{\mathrm{D}}$, and time-window $k \in \mathcal{K}_t$, with the

integer variable $x_{i,j,d,k,t}$. Naturally, the decision x_t is bounded by the feasible decision space $\mathcal{X}_t$ described by constraints (2b) to (2f).

$$x_t = [x_{i,j,d,k,t}]_{\forall (i,j) \in \mathcal{A}_t, d \in \mathcal{N}_t^{\mathrm{D}}, k \in \mathcal{K}_t} \tag{2a}$$

s.t.

$$\sum_{j \in \mathcal{N}_t^{\mathrm{I}} \cup \{d\}} x_{i,j,d,k,t} \leq F_{i,d,0,k,t}, \quad \forall i \in \mathcal{N}_t^{\mathrm{O}} \cup \mathcal{N}_t^{\mathrm{I}}, d \in \mathcal{N}_t^{\mathrm{D}}, k \in \mathcal{K}_t \tag{2b}$$

$$x_{i,d,d,L^A_{i,d,t},t} \geq F_{i,d,0,L^A_{i,d,t},t}, \quad \forall (i,d) \in \mathcal{A}_t^{\mathrm{D}}, k \in \mathcal{K}_t \tag{2c}$$

$$x_{i,j,d,k,t} = 0, \quad \forall (i,j) \in \mathcal{A}_t^{\mathrm{O}} \cup \mathcal{A}_t^{\mathrm{I}}, d \in \mathcal{N}_t^{\mathrm{D}}, k \in \mathcal{K}_t | k < M_{i,j,t} + \widetilde{M}_{j,d,t} \tag{2d}$$

$$\sum_{d \in \mathcal{N}_t^{\mathrm{D}}} \sum_{k \in \mathcal{K}_t} x_{i,j,d,k,t} \leq Q_{i,j,t}, \quad \forall (i,j) \in \mathcal{A}_t^{\mathrm{I}} \tag{2e}$$

$$x_{i,j,d,k,t} \in \mathbb{N} \cup \{0\}, \quad \forall (i,j) \in \mathcal{A}_t, d \in \mathcal{N}_t^{\mathrm{D}}, k \in \mathcal{K}_t \tag{2f}$$

Constraints (2b) ensure that, for every origin and intermodal terminal, only freights that are released can be transported. Constraints (2c) guarantee that freights whose time-window is as long as the duration of direct transportation (i.e., trucking) are transported using this service. Note that with this constraint, we assume that trucking to a destination is faster than going via an intermodal terminal, i.e., $L^A_{i,d,t} < \min_{j \in \mathcal{N}_t^{\mathrm{I}}} \left\{ M_{i,j,t} + L^A_{j,d,t} \right\}, \forall (i,d) \in \mathcal{A}_t^{\mathrm{D}}$. Equations (2d) ensure that freights are not transported to a terminal j if the fastest "intermodal" route to their destination after arriving at that terminal (whose duration we denote with $\widetilde{M}_{j,d,t}$), is longer than the freight's time-window. This strict definition of transportation options means that two trucking services cannot be used sequentially (e.g., bring a freight from its origin to an intermodal terminal by truck and then transport it with truck from that intermodal terminal to its destination). The value of $\widetilde{M}_{j,d,t}$ is case dependent: (i) if a freight is at an origin (i.e., $i \in \mathcal{N}_t^{\mathrm{O}}$), then $\widetilde{M}_{j,d,t}$ includes the duration of the shortest service from terminal j to terminal j' and the duration of the trucking from j' to the destination d; (ii) if a freight is at an intermodal terminal (i.e., $i \in \mathcal{N}_t^{\mathrm{I}}$) then $\widetilde{M}_{j,d,t} = M_{j,d,t}$ since the intermodal service $(i,j) \in \mathcal{A}_t^{\mathrm{I}}$ already accounts for the intermodal part. Constraints (2e) ensure that the capacity of each service is not exceeded. Finally, constraints (2f) define the domain of the variables.

After making a decision x_{t-1} and before entering the state S_t, exogenous information on new freights arrives. We denote the number of new freights with origin $i \in \mathcal{N}_t^{\mathrm{O}}$, destination $d \in \mathcal{N}_t^{\mathrm{D}}$, release day $r \in \mathcal{R}_t$, and time-window $k \in \mathcal{K}_t$ that arrive in between two consecutive stages $t-1$ and t with the integer variable $\widetilde{F}_{i,d,r,k,t}$. Hence, we model this exogenous information with the vector W_t, as seen in (3).

$$W_t = \left[\widetilde{F}_{i,d,r,k,t} \right]_{\forall i \in \mathcal{N}_t^{\mathrm{O}}, d \in \mathcal{N}_t^{\mathrm{D}}, r \in \mathcal{R}_t, k \in \mathcal{K}_t} \tag{3}$$

The transition from state S_{t-1} to state S_t depends on (i) the decision x_{t-1}, (ii) the exogenous information W_t, and (iii) the various time relations involving freights and services. We capture this transition using the function S^M, as seen in (4). First, and most naturally, decisions shift freights from one location to another through time. However, this shift can take longer than one stage,

i.e., when a service duration spans more than one day. To avoid remembering decisions of services spanning more than one day (i.e., earlier decisions than x_{t-1}), we use the virtual time-windows. As exemplified before, virtual time-windows increase the release-day and reduce the time-window of freights that are transported using a service with a duration longer than one day, i.e., $M_{i,j,t} > 1$. Second, the exogenous information increases the number of freights of a certain type that are present in the network. Third, various time relations apply to different types of freight. For example, released freights that are not transported remain at the same location and their time-window decreases. To capture all these relations, S^M categorizes freight variables $F_{t,i,d,r,k}$ into seven equations, as shown in (4b) to (4h). We now elaborate on each category specifically.

$$S_t = S^M\left(S_{t-1}, x_{t-1}, W_t\right) \tag{4a}$$

s.t.

$$F_{i,d,0,k,t} = F_{i,d,0,k+1,t-1} - \sum_{j \in \mathcal{A}_t} x_{i,j,d,k+1,t-1} + F_{i,d,1,k,t-1} + \widetilde{F}_{i,d,0,k,t}, \tag{4b}$$
$$\forall i \in \mathcal{N}_t^{\mathrm{O}}, d \in \mathcal{N}_t^{\mathrm{D}}, k+1 \in \mathcal{K}_t$$

$$F_{i,d,0,K_t^{\max},t} = F_{i,d,1,K_t^{\max},t-1} + \widetilde{F}_{i,d,0,K_t^{\max},t}, \tag{4c}$$
$$\forall i \in \mathcal{N}_t^{\mathrm{O}}, d \in \mathcal{N}_t^{\mathrm{D}}$$

$$\begin{aligned} F_{i,d,0,k,t} = F_{i,d,0,k+1,t-1} &- \sum_{j \in \mathcal{A}_t} x_{i,j,d,k+1,t-1} + F_{i,d,1,k,t-1} \\ &+ \sum_{j \in \mathcal{A}_t | M_{j,i,t}=1} x_{j,i,d,k+M_{j,i,t},t-1}, \end{aligned} \tag{4d}$$
$$\forall i \in \mathcal{N}_t^{\mathrm{I}}, d \in \mathcal{N}_t^{\mathrm{D}}, k+1 \in \mathcal{K}_t$$

$$F_{i,d,r,k,t} = F_{i,d,r+1,k,t-1} + \widetilde{F}_{i,d,r,k,t}, \tag{4e}$$
$$\forall i \in \mathcal{N}_t^{\mathrm{O}}, d \in \mathcal{N}_t^{\mathrm{D}}, r+1 \in \mathcal{R}_t | r \geq 1, k \in \mathcal{K}_t$$

$$F_{i,d,R_t^{\max},k,t} = \widetilde{F}_{i,d,R_t^{\max},k,t}, \tag{4f}$$
$$\forall i \in \mathcal{N}_t^{\mathrm{O}}, d \in \mathcal{N}_t^{\mathrm{D}}, k \in \mathcal{K}_t$$

$$F_{i,d,r,k,t} = F_{i,d,r+1,k,t-1} + \sum_{j \in \mathcal{A}_t | M_{j,i,t}=r+1} x_{j,i,d,k+M_{j,i,t},t-1}, \tag{4g}$$
$$\forall i \in \mathcal{N}_t^{\mathrm{I}}, d \in \mathcal{N}_t^{\mathrm{D}}, r+1 \in \mathcal{R}_t' | r \geq 1, k \in \mathcal{K}_t$$

$$F_{i,d,|\mathcal{R}_t'|,k,t} = \sum_{j \in \mathcal{A}_t | M_{j,i,t}=|\mathcal{R}_t'|+1} x_{j,i,d,k+M_{j,i,t},t-1}, \tag{4h}$$
$$\forall i \in \mathcal{N}_t^{\mathrm{I}}, d \in \mathcal{N}_t^{\mathrm{D}}, k \in \mathcal{K}_t$$

Equations (4b) define freights that are released (i.e., $r = 0$), at an origin (i.e, $i \in \mathcal{N}_t^{\mathrm{O}}$), and with a time-window smaller than $K_t^{\max}$ (i.e., $k+1 \in \mathcal{K}_t$), as the sum of two types of freights from the previous stage at that origin with the same destination: (i) freights with a time-window of one stage longer that were not transported (i.e., $F_{t-1,i,d,0,k+1} - \sum_{j \in \mathcal{A}_t} x_{t-1,i,j,d,k+1}$), (ii) freights that had a release-day of one (i.e., $F_{t-1,i,d,1,k}$) meaning that they would be released at the current stage, in addition to the new arriving freights in between the stages that had the same characteristics (i.e., $\widetilde{F}_{t,i,d,0,k}$). Equations (4c) define freights that are released, at an origin, and have the maximum time-window

as the sum of freights with a release-day of one and the new arriving freights with the same characteristics. Equations (4d) define freights that are released, at an intermodal terminal, and with a time-window smaller than the maximum one, as the result of three types of freights from the previous stage at that terminal with the same destination: (i) freights with a time-window of one stage longer that were not transported, (ii) freights that had a release-day of one, and (iii) inbound freights from all locations j whose service duration was one period (i.e., $M_{j,i,t} = 1$) and whose time-window was the current freight's time-window plus the service duration (i.e., a reduced time-window from $k + M_{j,i,t}$ to k) at the moment of the decision x_{t-1} (i.e., $\sum_{j \in \mathcal{A}_t | M_{j,i,t}=1} x_{t-1,j,i,d,k+M_{j,i,t}}$). Remind that, due to our separation of nodes, no new arriving freights come to intermodal terminals and that, due to our assumption of a minimum of one day service duration, no freights at intermodal terminals can have the maximum time-window. Equations (4e) define freights at an origin node that are still not released and do not have the maximum release-day, as the sum of two types of freight from the previous stage at that origin with the same destination and time-window: (i) freights with a release-day of one period longer than the current freight's release-day, and (ii) new arriving freights that had the same characteristics. Freights of the previous type that have the maximum release-day are the result of only the new arriving freights with the same characteristics, as shown in Equations (4f). Equations (4g) define freights that are at an intermodal terminal and that are not released but do not have the maximum release-day, as the sum of two types of freight from the previous stage at that terminal: (i) freights with a release-day of one period longer than the current freight's release-day and the same time-window, and (ii) freights sent in the decision of the previous stage, from all other locations to that terminal, whose service duration was equal to the release-day of the current freight plus one period and whose time-window was equal to the current freight's time-window plus the aforementioned service duration. Finally, freights at an intermodal terminal with the maximum release-day of the virtual time-windows are the result of inbound freights to that location following the virtual time-windows reasoning, as shown in Equations (4h).

The immediate rewards of a decision $R_t(x_t)$ are calculated as shown in Equation (5a). Remind that $A_{i,j,d,t}$ and $C_{i,j,d,t}$ are the revenue and variable cost of using service $(i,j) \in \mathcal{A}_t$ for one freight with destination d, respectively, and that $B_{i,j,t}$ is the setup cost for using the aforementioned service independent of the number of freights.

$$R_t(x_t) = \sum_{(i,j)\in\mathcal{A}_t} \sum_{d\in\mathcal{N}_t^{\mathrm{D}}} \left((A_{i,j,d,t} - C_{i,j,d,t}) \sum_{k\in\mathcal{K}_t} x_{i,j,d,k,t} \right) - \sum_{(i,j)\in\mathcal{A}_t} (B_{i,j,t} \cdot y_{i,j,t}) \tag{5a}$$

where

$$y_{i,j,t} = \begin{cases} 1, & \text{if } \sum_{d\in\mathcal{N}_t^{\mathrm{D}}} \sum_{k\in\mathcal{K}_t} (x_{i,j,d,k,t}) > 0 \\ 0, & \text{otherwise} \end{cases}, \ \forall (i,j) \in \mathcal{A}_t \tag{5b}$$

Since the decision x_t depends on the state, and the state is partially random, the objective is to find a policy that maximizes the expected discounted reward

over the planning horizon. We denote a policy with $\pi \in \Pi$, and define it as a function that determines a decision $x_t^\pi \in \mathcal{X}_t$ for each possible state $S_t \in \mathcal{S}$. Thus, the objective can be expressed as shown in (6), where γ_t is the discount factor balancing the impact of future and present rewards and S_0 is the initial state of the system.

$$\max_{\pi \in \Pi} \mathbb{E}\left[\sum_{t \in \mathcal{T}} \gamma_t R_t\left(x_t^\pi\right) \middle| S_0\right] \tag{6}$$

Finite horizon MDP models such as ours can be solved using linear programming or dynamic programming. To design our solution approach, we focus on the latter. In dynamic programming, the optimal expected rewards can be estimated using a set of recursive equations and Bellman's principle of optimality, as shown in Equations (7). These equations can be solved backwards, from the end of the horizon towards the beginning. In Equations (7a), the state S_{t+1} is partially random and partially the result of decision x_t. Using the transition function S^M, we can express S_{t+1} as a function of the current state, decision, and a realization of the exogenous information, as shown in Equations (7b). Assuming that the possible realizations of the exogenous information Ω_{t+1} (i.e., $W_{t+1} \in \Omega_{t+1}$) are finite, and defining $p_\omega^{\Omega_{t+1}}$ as the probability of realization $\omega \in \Omega_{t+1}$, we can recursively solve our MDP as shown in Equations (7c). The solution to these last equations yield the optimal policy π^*. However, solving these equations is challenging for various reasons. In the following, we elaborate further on those solution challenges.

$$V_t\left(S_t\right) = \max_{x_t \in \mathcal{X}_t}\left(R_t\left(x_t\right) + \gamma_t \mathbb{E}\left[V_{t+1}\left(S_{t+1}\right)\right]\right), \quad \forall S_t \in \mathcal{S} \tag{7a}$$

$$V_t\left(S_t\right) = \max_{x_t \in \mathcal{X}_t}\left(R_t\left(x_t\right) + \gamma_t \mathbb{E}\left[V_{t+1}\left(S^M\left(S_t, x_t, W_{t+1}\right)\right)\right]\right), \quad \forall S_t \in \mathcal{S} \tag{7b}$$

$$V_t\left(S_t\right) = \max_{x_t \in \mathcal{X}_t}\left(R_t\left(x_t\right) + \gamma_t \sum_{\omega \in \Omega_{t+1}} p_\omega^{\Omega_{t+1}}\left(V_{t+1}\left(S^M\left(S_t, x_t, \omega\right)\right)\right)\right), \quad \forall S_t \in \mathcal{S} \tag{7c}$$

4.3 Solution Challenges

As with most MDP models, ours suffers from the so-called three curses of dimensionality [27]. Equations (7c) hold for the entire state space $\mathcal{S}$, decision space $\mathcal{X}_t$, and exogenous information space Ω_t, which grow larger with an increasing size of the transportation network and demand parameters. In addition to the curses of dimensionality, the assumption of a finite horizon in our MDP model brings additional challenges compared to an infinite horizon MDP. For example, in an infinite horizon MDP, the resulting policy is independent of the stage opposed to the finite horizon where it depends on the stage. In other words, infinite horizon MDPs yield one policy instead of $T^{\max}$ policies in the finite case. Even if we are only interested in the policy for the first decision moment in the finite case, we still need to learn the values for $|\mathcal{S}|$ states for $T^{\max}$ stages. However, the stationary-information assumption of the infinite horizon MDPs is not desired in our problem, where freight demand can have seasonality effects (e.g., different demand distributions on different stages) and services can change (e.g., increased capacity of a train, disruptions, etc.) over time. Although this stage-dependent information can be added to the state and transition definition

in an infinite horizon, at the price of a larger state space, the finite horizon MDP allows us to start from the current state and use up-to-date information of what might happen in the finite period that lies ahead. In other words, it is easier to use the MDP for re-planning purposes once the information changes, which is a key characteristic of synchromodality.

Another solution challenge arises due to the interaction among the model's reward function, transition function, and the finite horizon; and the need for heuristic/approximation solution methodologies. In the freight transportation industry, the revenue of a freight is usually received at a single point in time (e.g., at the pick-up or at the delivery) while the costs for the entire route are accrued in several points in time depending on the route. Although this is not an issue for the MDP model, it may result in strange behavior of heuristic approaches. When revenue is received at the pick-up, heuristics might prefer to pick up a freight as soon as possible, to receive the revenue, and then take it to the closest location and leave it there in order to avoid costs. In the opposite case, when the revenue is received at the delivery, heuristics might prefer to take a freight to its destination, as soon as possible, to receive the revenue, and avoid longer transportation options which might be cheaper. These two behaviors accentuate more when considering the end-of-the-horizon effects. When revenue is received at the pick-up at the origin, heuristics might tend to postpone the transportation of freight till the end of the horizon. When revenue is received at the delivery to the destination, heuristics might aim for an empty network at the end of the horizon. Consequently, the "greedy" nature of heuristics may lead to a poor performance in our finite horizon look-ahead model.

Although our MDP model brings various solution challenges, its components (e.g., transition function, decision constraints, exogenous information) can be used within the approximate dynamic programming framework to design a heuristic solution for them. We elaborate on this design, and how to overcome the challenges, in the following section.

5 Approximate Dynamic Programming Solution

To solve the MDP model from the previous section, we use a heuristic approach based on the framework of Approximate Dynamic Programming (ADP). In ADP, the solution to the Bellman's Equations in (7) is approximated using simulation, along with other optimization and statistical techniques, in an iterative manner. In this section, we elaborate on two ADP designs. In Section 5.1, we describe a traditional design using *basis functions* and ϵ-greedy exploration, that has worked in other intermodal transportation problems [26]. Furthermore, we comment on how the characteristics of our problem can make this traditional design flounder in apparent local-optima. In Section 5.2, we present ways to avoid these local-optima using ideas from Bayesian exploration and implementing them into a hybrid ADP design. Before elaborating on the traditional and hybrid ADP designs, we briefly describe the workings of the ADP algorithm. For an in-depth explanation of the ADP algorithm and its parts, we refer to [27].

The ADP algorithm consists of various parts of the MDP model, such as the state, decision, and transition function definition, as shown in Algorithm 1. The ADP algorithm runs for a number of iterations N and hence has all its variables

Algorithm 1 ADP Algorithm

1: Initialize $\left[\overline{V}_t^0\right]_{\forall t \in \mathcal{T}}$
2: **for** $n = 1$ **to** N **do**
3: $\quad S_0^n := S_0$
4: $\quad$ **for** $t = 0$ **to** $T^{max} - 1$ **do**
5: $\quad\quad x_t^{n*} := \underset{x_t^n \in \mathcal{X}_t^{\mathrm{R}}}{\arg\max} \left(R_t\left(x_t^n\right) + \gamma_t \overline{V}_t^{n-1}\left(S^{M,x}\left(S_t^n, x_t^n\right)\right)\right)$
6: $\quad\quad S_t^{n,x*} := S^{M,x}\left(S_t^n, x_t^{n*}\right)$
7: $\quad\quad \widehat{v}_t^n := \left(R_t\left(x_t^{n*}\right) + \gamma_t \overline{V}_t^{n-1}\left(S_t^{n,x*}\right)\right)$
8: $\quad\quad W_{t+1}^n := \mathrm{Random}\left(\Omega\right)$
9: $\quad\quad S_{t+1}^n := S^M\left(S_t^n, x_t^{n*}, W_{t+1}^n\right)$
10: $\quad$ **end for**
11: $\quad$ **for** $t = T^{max} - 1$ **to** 0 **do**
12: $\quad\quad \overline{V}_t^n(S_t^{n,x*}) := U_t^n(\overline{V}_t^{n-1}(S_t^{n,x*}), S_t^{n,x*}, [\widehat{v}_t^n]_{\forall t \in \mathcal{T}})$
13: $\quad$ **end for**
14: **end for**
15: **return** $\left[\overline{V}_t^N\right]_{\forall t \in \mathcal{T}}$

indexed with n, as shown in Line 2. The overall idea of ADP is to replace the expectation of future rewards in (7a) with an approximation $\overline{V}_t^n$ and to update this approximation over the iterations. At the end, the algorithm yields the approximation of the last iteration, for all stages, as shown in Line 15. Thus, the output of ADP is a decision-making policy, based on the approximation $\left[\overline{V}_t^N\right]_{\forall t \in \mathcal{T}}$ of the expected rewards, in a similar way to the MDP.

The approximation $\overline{V}_t^n$ is a function of the so-called *post-decision state* $S_t^{x,n}$. The post-decision state is the state of the system after a decision has been made but before the new exogenous information becomes known and the next stage state is realized, i.e., $S_t^{n,x} = S^{M,x}\left(S_t^n, x_t^n\right)$. The transition $S^{M,x}$ to the post-decision state works in a similar way as the transition function S^M, see (4), with the only difference that no exogenous information is considered, i.e., all $\widetilde{F}_{i,d,r,k,t}$ variables are omitted in (4b) to (4h). To update the approximation $\overline{V}_t^n$, the algorithm simulates the use of its resulting policy for all stages. In contrast to backwards dynamic programming, ADP moves forward in the stages, as shown in Line 4. In each stage, the optimality equations in (7) are transformed into one single equation (using $\overline{V}_t^n$), as shown in Line 7. Furthermore, the decision that attains the maximum reward $\widehat{v}_t^n$, as well as its corresponding post-decision state, are stored as shown in Lines 5 and 6. To advance to the next stage $t+1$, the algorithm uses a sample from Ω_{t+1}, obtained through a Monte Carlo simulation, and the transition function S^M defined in (4), as shown in Line 8. After all stages are processed, a function U_{t-1}^n is used to update the approximation in a backward manner, as seen in Lines 11 to 13. This function uses the information stored throughout the stages. The entire procedure just described is then repeated N times.

The benefit of having the approximation $\overline{V}_t^n$ in ADP is two-fold. First, it avoids enumerating all possible realizations of the exogenous information Ω_t. Second, it allows the optimality equation in Line 7 to be solved for one state

at a time, instead of for all states. These two benefits eliminate two of the curses of dimensionality. However, the large decision space must be tackled separately. For this, we propose the use of a *Restricted Policy* (RP). We design two RPs by adding more constraints to the feasible decision space $\mathcal{X}_t$ described by constraints (2b) to (2f). All restrictions significantly reduce the number of feasible decisions to evaluate; some restrictions do so without harming solution quality whereas others limit the solution quality that ADP can attain. We elaborate more on this issue in Section 6.4. For now, we describe the two RPs.

The first RP, denoted by $\mathcal{X}_t^{\text{RP1}}$ and defined in (8), works with three additional constraints to the feasible decision space. First, freights that are not urgent (i.e., $k > L^A_{i,d,t}$) cannot use direct trucking to their destination (i.e., service (i,d)), as shown in Restriction (8b). Second, all freights that are at the same location and go to the same destination must be transported together, as shown in Restriction (8c). To achieve this, we use the binary variable $x^{\text{RG}}_{i,j,d,t}$, which gets a value of 1 if freights at location i with destination d are sent using service $(i,j) \in \mathcal{A}^{\text{I}}_t$ and 0 otherwise, and the binary parameter $M^{\text{R}}_{i,j,d,k,t}$, which gets a value of 1 if the fastest intermodal route for freights at location i with destination d and time-window k is longer than the freight's time-window (i.e., $k < M_{i,j,t} + \widetilde{M}_{i,d,t}$). Third, we consider that all freights that arrive at an origin, independent of their characteristics, must be either transported to the same intermodal terminal or postponed, as shown in Restriction (8d). To achieve this, we use the binary variable $x^{\text{RO}}_{j,t}$, which gets a value of 1 if the chosen intermodal terminal is j and 0 otherwise. Naturally, we assume with this last restriction that there is at least one terminal to which freight from all origins can be brought to.

$$x_t \in \mathcal{X}_t \tag{8a}$$

$$x_{i,d,d,k,t} = 0, \quad \forall i \in \mathcal{N}^{\text{O}}_t \cup \mathcal{N}^{\text{I}}_t, d \in \mathcal{N}^{\text{D}}_t, k \in \mathcal{K}_t | k > L^A_{i,d,t} \tag{8b}$$

$$x_{i,j,d,k,t} \geq (F_{i,d,0,k,t}) \left(x^{\text{RG}}_{i,j,d,t} - M^{\text{R}}_{i,j,d,k,t}\right), \quad \forall i \in \mathcal{N}^{\text{O}}_t \cup \mathcal{N}^{\text{I}}_t, j \in \mathcal{N}^{\text{I}}_t, d \in \mathcal{N}^{\text{D}}_t, k \in \mathcal{K}_t \tag{8c}$$

$$\sum_{d \in \mathcal{N}^{\text{D}}_t} \sum_{i \in \mathcal{N}^{\text{O}}_t} x_{i,j,d,k,t} = \left|\mathcal{N}^{\text{D}}_t\right| \left|\mathcal{N}^{\text{O}}_t\right| x^{\text{RO}}_{j,t}, \quad \forall j \in \mathcal{N}^{\text{I}}_t | \exists_{\forall i \in \mathcal{N}^{\text{O}}_t} (i,j) \in \mathcal{A}^{\text{I}}_t \tag{8d}$$

The second RP, denoted by $\mathcal{X}_t^{\text{RP2}}$ and defined in (9), works similarly to RP1 with one difference. Instead of grouping all freights that arrive in all origins, as in (8d), we group all freights that arrive in all origins that go to the same destination, as seen in (9d). This means that all freight that arrives to any origin with a given destination must either be brought to the same terminal or postponed. To achieve this, we use the binary variable $x^{\text{RO2}}_{j,d,t}$, which gets a value of 1 if the intermodal terminal j is chosen for freights with destination d and 0 otherwise. RP2 is computationally more expensive than RP1 but also gives more options to choose from at the pick-up of freight, possibly resulting in better solutions. We analyze the solution quality of both RPs in the Section 6. It is important to note that, if computational resources allow, these restricted policies can be omitted.

$$x_t \in \mathcal{X}_t \tag{9a}$$

$$x_{i,d,d,k,t} = 0, \quad \forall i \in \mathcal{N}^{\text{O}}_t \cup \mathcal{N}^{\text{I}}_t, d \in \mathcal{N}^{\text{D}}_t, k \in \mathcal{K}_t | k > L^A_{i,d,t} \tag{9b}$$

$$x_{i,j,d,k,t} \geq \left(F_{i,d,0,k,t}\right)\left(x^{\mathrm{RG}}_{i,j,d,t} - M^{\mathrm{R}}_{i,j,d,k,t}\right), \tag{9c}$$
$$\forall i \in \mathcal{N}^{\mathrm{O}}_t \cup \mathcal{N}^{\mathrm{I}}_t, j \in \mathcal{N}^{\mathrm{I}}_t, d \in \mathcal{N}^{\mathrm{D}}_t, k \in \mathcal{K}_t$$

$$\sum_{i \in \mathcal{N}^{\mathrm{O}}_t} x_{i,j,d,k,t} = \left|\mathcal{N}^{\mathrm{O}}_t\right| x^{\mathrm{RO2}}_{j,d,t}, \quad \forall j \in \mathcal{N}^{\mathrm{I}}_t | \exists_{\forall i \in \mathcal{N}^{\mathrm{O}}_t}(i,j) \in \mathcal{A}^{\mathrm{I}}_t, d \in \mathcal{N}^{\mathrm{D}}_t \tag{9d}$$

5.1 ADP with Basis Functions and Epsilon-Greedy Exploration

In our first ADP design, we use *basis functions* for the approximation $\overline{V}^n_t(S^{x,n}_t)$. The overall idea of basis functions is to quantify characteristics of a post-decision state that explain the expected future rewards to a certain degree. We denote the basis function of a characteristic $b \in \mathcal{B}_t$ with $\phi_{b,t} : S^{x,n}_t \rightarrow \mathbb{R}$ and the degree (i.e., weight) with which it explains the future rewards by $\theta^n_{b,t} \in \mathbb{R}$. The approximated future rewards are the result of the product between all basis functions and their weights, as shown in (10). Using matrix notation, we define $\phi_t\left(S^{x,n}_t\right) = \left[\phi_{1,t}\left(S^{x,n}_t\right), ..., \phi_{|\mathcal{B}_t|,t}\left(S^{x,n}_t\right)\right]^{\mathrm{T}}$ as the column vector of basis functions, and $\theta^n_t = \left[\theta^n_{1,t}, ..., \theta^n_{|\mathcal{B}|,t}\right]^{\mathrm{T}}$ as the column vector of weights at iteration n and stage t. We will use these vectors to describe the updating procedure later on.

$$\overline{V}^n_t\left(S^{x,n}_t\right) = \sum_{b \in \mathcal{B}} \theta^n_{b,t} \phi_{b,t}\left(S^{x,n}_t\right) = \phi_t\left(S^{x,n}_t\right)^{\mathrm{T}} \theta^n_t \tag{10}$$

At each stage t, the set of characteristics $\mathcal{B}_t$ counts two types of freights per location-destination pair: (i) freights whose time-window is shorter than the duration of the shortest intermodal path of the entire network, which we denote as ψ, as shown in (11a), and (ii) freights whose time-window is at least the duration of the shortest intermodal path of the entire network, as shown in (11b). Furthermore, we also count the total number of freights going to each destination, independent of their current location, release-day, or time-window, as shown in (11c). Finally, we add a constant as shown in (11d). We have a total number of characteristics $|\mathcal{B}_t| = 2\left(\left|\mathcal{N}^{\mathrm{O}}_t\right|\left|\mathcal{N}^{\mathrm{I}}_t\right|\right) + \left|\mathcal{N}^{\mathrm{D}}_t\right| + 1$. To index them, we use the functions b, b'', and b''', with range $[1, ..., |\mathcal{B}_t|]$.

$$\phi_{b(i,d),t}\left(S^{x,n}_t\right) = \sum_{k \in \mathcal{K}_t | k < \Psi} \sum_{r \in \mathcal{R}'_t} F^{x,n}_{i,d,r,k,t}, \quad \forall i \in \mathcal{N}^{\mathrm{O}}_t \cup \mathcal{N}^{\mathrm{I}}_t, d \in \mathcal{N}^{\mathrm{D}}_t \tag{11a}$$

$$\phi_{b'(i,d),t}\left(S^{x,n}_t\right) = \sum_{k \in \mathcal{K}_t | k \geq \Psi} \sum_{r \in \mathcal{R}'_t} F^{x,n}_{i,d,r,k,t}, \quad \forall i \in \mathcal{N}^{\mathrm{O}}_t \cup \mathcal{N}^{\mathrm{I}}_t, d \in \mathcal{N}^{\mathrm{D}}_t \tag{11b}$$

$$\phi_{b''(d),t}\left(S^{x,n}_t\right) = \sum_{i \in \mathcal{N}^{\mathrm{O}}_t \cup \mathcal{N}^{\mathrm{I}}_t} \sum_{k \in \mathcal{K}_t | k \geq \Psi} \sum_{r \in \mathcal{R}'_t} F^{x,n}_{i,d,r,k,t}, \quad \forall d \in \mathcal{N}^{\mathrm{D}}_t \tag{11c}$$

$$\phi_{|\mathcal{B}_t|}\left(S^{x,n}_t\right) = 1 \tag{11d}$$

The basis functions $\phi_t(S^{x,n}_t)$ are fixed for all iterations, changing value only with respect to the post-decision state $S^{x,n}_t$ and the network at stage t. The weights $\theta^n_{b,t}$, however, depend on an iteration n and stage t. The idea is that, throughout the iterations, the observed rewards for each stage can be used to update the weights, and thus the approximation $\overline{V}^n_t(S^{x,n}_t)$. For the updating function U^n_{t-1}, we use recursive least squares for non-stationary data. This method uses three variables to update the weights: (i) the difference between the

next-stage estimate from the previous iteration $\overline{V}_{t-1}^{n-1}\left(S_{t-1}^{x,n}\right)$ and the observed reward $\widehat{v}_t^n$ at the current iteration, (ii) the value of each basis function $\phi_b\left(S_t^{x,n}\right)$, and (iii) the optimization matrix H^n, as shown in (12). This optimization matrix can put more emphasis on recent observations. This can be beneficial when in early iterations the approximation is far from optimal or when the initial conditions of the system may bias the approximation. For a comprehensive explanation on the recursive least squares method for non-stationary data we refer to [27].

$$\overline{V}_t^n(S_t^{n,x*}) := U_t^n(\overline{V}_t^{n-1}(S_t^{n,x*}), S_t^{n,x*}, [\widehat{v}_t^n]_{\forall t \in \mathcal{T}}) \tag{12a}$$

s.t.

$$\theta_t^n = \theta_t^{n-1} - \left(H_t^n\right)^{-1} \phi_t\left(S_t^{x,n}\right) \left(\overline{V}_{t-1}^{n-1}\left(S_{t-1}^{x,n}\right) - \sum_{t}^{T^{\max}-1} \widehat{v}_t^n \right) \tag{12b}$$

$$H_t^n = \lambda^n H_t^{n-1} + \phi_t\left(S_t^{x,n}\right) \phi_t\left(S_t^{x,n}\right)^{\mathrm{T}} \tag{12c}$$

$$\lambda^n = 1 - \frac{\lambda}{n} \tag{12d}$$

The ADP design presented before uses the so-called *exploitation* decision strategy. Exploitation decisions are those that take us to the best post-decision state given our estimate of the downstream rewards at that iteration, as seen in Line 5 of Algorithm 1. If these estimates are far from good, the post-decision state they take us too can also be far from optimal, and since the estimates are updated with the post-decision state we saw (i.e., using the basis functions), we might end-up in a chain reaction for the worse. A different approach to make decisions, which aims to avoid such cycles during the learning phase of ADP, is to consider *exploration* decisions. As their name states, exploration decisions are those that take us to post-decision states that we might not have seen before, even if they are not the best ones given. In analogy to local search heuristics, exploration decisions can be seen as moves that prevent ADP from getting stuck in local optima. The benefit of making exploration decisions in our basis functions approach is that we may observe post-decision state characteristics we had not seen before (i.e., basis function values different than zero), or not that often, and therefore improving the approximation of the downstream rewards. One way to consider exploration decisions is the ϵ-greedy strategy [27]. In this strategy, a fraction ϵ of the decisions through the iterations should be exploration ones. In other words, a random decision from $\mathcal{X}_t^{\mathrm{R}}$ is chosen with probability ϵ. To implement it, Line 5 in Algorithm 1 must be exchanged with Algorithm 2.

Algorithm 2 ϵ-greedy strategy for exploration

1: **if** Random $[0, 1) < \epsilon$ **then**
2: $\quad x_t^{n*} :=$ Random $\left(\mathcal{X}_t^{\mathrm{R}}\right)$
3: **else**
4: $\quad x_t^{n*} := \underset{x_t^n \in \mathcal{X}_t^{\mathrm{R}}}{\arg\max} \left(R_t\left(x_t^n\right) + \gamma_t \bar{V}_t^{n-1}\left(S^{M,x}\left(S_t^n, x_t^n\right)\right)\right)$
5: **end if**

Although exploration decisions may take us to possibly better post-decision states, we also run the risk of deteriorating our approximation and thus making

worse exploitation decisions. This is caused by updating the weights of basis functions that we have already seen (for which we may have reasonable values) using values resulting from possibly far-from-optimal decisions. Although one could update the approximation using the exploitation decision rather than the exploration one (known as off-policy updating), in a finite horizon problem this results in a larger propagation of error across the stages since the observed post-decision states (and hence the value of the basis functions we use for updating our approximation) depend on the exploration decision and not the exploitation one. In the following section, we introduce a different exploration strategy that aims to balance the trade-off between going to relatively unknown post-decision states and deteriorating the current approximation.

5.2 ADP with Bayesian Exploration

Exploration decisions have the potential to improve our approximation when the approximation is not good. Exploitation decisions have the potential to improve our approximation further once the approximation is good. Since there is uncertainty on whether we already have a good approximation, the balance between exploration and exploitation decisions can be seen as a stochastic optimization sub-problem. A way to quantify the uncertainty and balance this tradeoff in ADP is through Bayesian exploration, using the concept of Value of Perfect Information (VPI), as proposed in Ryzhov et al. [33]. Although this technique has been recently applied successfully to infinite horizon problems, applying it to finite horizon problems such as ours comes with several challenges. As far as we know, this has not been considered before. In this Section, we introduce the general idea of Bayesian exploration and subsequently present the challenges and possible modifications to apply it to finite horizon problems in what we call a hybrid ADP design. For in-depth explanation of Bayesian exploration we refer the reader to Powell and Ryzhov [28].

During the early iterations of the ADP algorithm, there is a lot of uncertainty about the approximation of the downstream rewards. As the algorithm progresses, and more post-decision states are observed, this uncertainty is reduced. However, if the same post-decision states are observed over and over again, there could be a bias towards these post-decision states just because we have good estimates about their downstream rewards (i.e., due to updating the corresponding basis function weights). The general idea of Bayesian exploration is to prevent such a bias in the algorithm by making decisions that will provide information about which post-decision states are actually better than the ones we thought were best. In contrast to the way of making decisions in ϵ-greedy exploration, in Bayesian exploration we make the decision that provides the maximum *value of exploration* $v_t^{E,n}$ at stage t and iteration n, as shown in (13). In (13), K_t^n represents the knowledge about the uncertainty in the approximation of the downstream rewards. We now describe $v_t^{E,n}$ and K_t^n in more detail.

$$x_t^{n*} = \underset{x_t^n \in \mathcal{X}_t^{\mathrm{R}}}{\arg\max} \left(v_t^{E,n}(K_t^n, S_t^n x_t^n) \right) \tag{13}$$

In the VPI concept applied to ADP, the value of exploration $v_t^{E,n}$ of a decision x_t^n is defined as the expected improvement in the approximated downstream

reward that arises from visiting the post-decision state corresponding to x_t^n. It is defined as "expected" because the true value of the approximation is considered to be a random variable for which we have an initial distribution of belief [28]. The best estimate of the mean of this random variable at iteration n and stage t is $\overline{V}_t^n(S_t^{n,x})$. Recall that in our basis function design, $\overline{V}_t^n(S_t^{n,x})$ is the scalar product of the values of the basis functions for post-decision state $S_t^{n,x}$ and the weights. Thus, the variance of $\overline{V}_t^n(S_t^{n,x})$ is dependent on the weights θ_t^n and the basis functions ϕ_t. We define C_t^n as the $|\mathcal{B}_t|$ by $|\mathcal{B}_t|$ covariance matrix at iteration n and stage t. The uncertainty knowledge K_t^n of the approximated downstream rewards is defined as the tuple shown in (14).

$$K_t^n = (\overline{V}_t^n, C_t^n) = (\phi_t, \theta_t^n, C_t^n) \tag{14}$$

The value of exploration of a decision should be larger for those decisions that lead us to choosing a better decision in future iterations, given that we are at the same state in the same stage, than for the decision that we currently think is the best. Furthermore, the value of exploration should decrease as we explore though the iterations since the uncertainty of the estimated downstream rewards is also reduced with increasing number of observations. Eventually, when we are confident of discriminating between optimal and non-optimal decisions, only exploitation decisions should be made in order to improve the approximation $\overline{V}_t^n(S_t^{x,n})$. To achieve this, VPI (as applied by Ryzhov et al. [33] in the context of infinite horizon ADP) builds upon the notion of the value of information in reinforcement learning systems [9] and defines $v_t^{E,n}$ using the elements of K_t^n as shown in (15). In (15), the function f quantifies the knowledge gain of an exploration decision based on the initial belief distribution, and is defined using $f(z) = z\Phi(z) + \varphi(z)$ where Φ is the cumulative distribution function and φ the probability density function of a standard Gaussian distribution. Furthermore, $\sigma_t^{2,n}(S_t^{x,n})$ represents the prior variance of $\overline{V}_t^n(S_t^{n,x})$ and is computed as shown in (15c). For the comprehensive description and derivation of (15), we refer the reader to Ryzhov et al. [33].

$$v_t^{E,n}(K_t^n, S_t^n, x_t^n) = \sqrt{\sigma_t^{2,n}(S_t^{x,n})} f\left(-\frac{\delta(S_t^{x,n})}{\sqrt{\sigma_t^{2,n}(S_t^{x,n})}}\right) \tag{15a}$$

s.t.

$$\delta(S_t^{x,n}) = \left|\overline{V}_t^{x,n}(S_t^{x,n}) - \max_{y_t^n \in \mathcal{X}_t | y_t^n \neq x_t^n} \overline{V}_t^{x,n}(S_t^{y,n})\right| \tag{15b}$$

$$\sigma_t^{2,n}(S_t^{x,n}) = \phi(S_t^{x,n})^T C_t^n \phi(S_t^{x,n}) \tag{15c}$$

In (15), we observe that the larger the uncertainty $\sigma_t^{2,n}(S_t^{x,n})$ about the impact of decision x_t^n, the larger the value of exploration is. Besides the uncertainty, the difference between the approximated downstream rewards of decision x_t^n (given by $\overline{V}_t^{x,n}(S_t^{x,n})$) and the best decision of the remaining feasible decisions $y_t^n \in \mathcal{X}_t | y_t^n \neq x_t^n$ (given by $\overline{V}_t^{x,n}(S_t^{y,n})$) is considered. The larger this difference is, the lower the value of exploration $v_t^{E,n}$ corresponding with decision x_t^n becomes. Thus, the definition of $v_t^{E,n}(K_t^n, S_t^n, x_t^n)$ in (15) has the desired characteristic of reducing the risk of deteriorating our approximation through exploration decisions. To implement it, Line 5 in Algorithm 1 must be

exchanged with (13). In addition, the covariance matrix C_t^n must be initialized for all stages $t \in \mathcal{T}$ before Line 2 in Algorithm 1. Since it is difficult to define an initial distribution of belief about the true value of the approximation, the covariance matrix is usually initialized with a large number χ^{C} on the diagonal and with zero on its other entries [32]. This initialization resembles the case of no prior knowledge about the relation between between the weights of the basis functions.

The approximation and the belief about its distribution must be updated after each iteration. To update the approximated downstream rewards (i.e., update the weights of the basis functions), we use (16) where $\widehat{v}_t^n$ is the value of the exploration decision in (13), as calculated in Line 7 in Algorithm 1, and $\sigma^{2,\mathrm{E}}$ is a noise term due to the measurement error. This updating procedure is identical to the analogous recursive least squares method [33], and considers the difference between approximated and observed downstream rewards as well as the current uncertainty knowledge through the covariance matrix. Naturally, besides using the observed rewards, the observed basis functions can be used to update the covariance matrix. Remind that, with increasing number of observations of a post-decision state $S_t^{x,n}$ (i.e., observed basis functions), the uncertainty about the approximated downstream rewards of that post-decision state decreases. We update the covariance matrix as shown in (17), again using the noise term $\sigma^{2,\mathrm{E}}$. To implement these updating methods, we replace Line 12 in Algorithm 1 with (16) and (17).

$$\theta_t^n = \theta_t^{n-1} - \frac{\left(\theta_t^{n-1}\right)^T \phi\left(S_t^{x,n}\right) - \sum_{t}^{T^{\max}-1} \widehat{v}_t^n}{\sigma^{2,\mathrm{E}} + \sigma_t^{2,n-1}(S_t^{x,n})} C_t^n \phi\left(S_t^{x,n}\right) \tag{16}$$

$$C_t^n = C_t^{n-1} - \frac{C_t^{n-1}\phi\left(S_t^{x,n}\right)\phi\left(S_t^{x,n}\right)^T C_t^{n-1}}{\sigma^{2,\mathrm{E}} + \sigma_t^{2,n}(S_t^{x,n-1})} \tag{17}$$

In the updating procedure explained above, we use the downstream rewards from the exploration decisions. This differs from Ryzhov et al. [33], where the approximated downstream rewards of the exploitation decision are being used instead of the exploration decision. The use of one decision, and its value, to update the approximation and a different decision in the transition of the system is known as off-policy updating, as mentioned before. Although off-policy updating can be useful to prevent exploration decisions harming the approximation in infinite horizon problems, such as the one of Ryzhov et al. [33], it is difficult to apply it to finite horizon problems, especially in combination with backwards updating [37, 38]. To prevent exploration decisions affecting our approximation in a negative way, we propose to be slightly more conservative in three aspects of exploration: (i) the definition of "gain" in the value of exploration, i.e., $\delta(S_t^{x,n})$ defined in (15b), (ii) the use of the value of exploration in making decisions, i.e., x_t^{n*} defined in (13), and (iii) the updates resulting from exploration decisions using the noise term $\sigma^{2,\mathrm{E}}$ as shown in (16) and (17). We now elaborate on each of these aspects.

5.2.1 The Gain in Value of Exploration

To be more conservative with the value of exploration $v_t^{E,n}$, we can incorporate more aspects of the exploitation decision. The first aspect we note is the exclu-

sion of the direct rewards $R_t\left(x_t^n\right)$ in the calculation of the value of exploration of decision x_t^n, as shown in (15). On the one hand, it is reasonable to use only the expected downstream rewards $\overline{V}_t^{x,n}\left(S_t^{x,n}\right)$ in (15) because the post-decision state corresponding to decision x_t^n and its basis functions might be observed without, or with different, direct rewards at a later iteration. Remind that different states and different decisions at stage t might still lead to the same post-decision state and thus the same basis function values. On the other hand, if observing basis function values would always involve direct rewards (as in the "online" use of ADP which we describe at the end of the next section), it makes sense to include them in the value of exploration. Thus, we can add $R_t\left(\cdot\right)$ to (15b), as seen in (18).

$$\delta(S_t^n, x_t^n) = \left| R_t\left(x_t^n\right) + \overline{V}_t^{x,n}\left(S_t^{x,n}\right) - \max_{y_t^n \in \mathcal{X}_t^{\mathrm{R}} | y_t^n \neq x_t^n} \left(R_t\left(y_t^n\right) + \overline{V}_t^{x,n}\left(S_t^{y,n}\right) \right) \right| \tag{18}$$

5.2.2 The Exploration Decision

Although the idea from the previous modification decreases the value of exploration for decisions with relatively low direct rewards, the exploration decision itself, as given by (13), is still solely based in the value of exploration. Another way to be conservative with the exploration decisions is to directly include, in addition to the value of exploration, some aspects of the exploitation decision (i.e., Line 5 in Algorithm 1). Naturally, there are many forms to include these aspects. We propose three forms of doing so.

First, we can include the approximated downstream reward when making a decision as proposed by Dearden et al. [9] and shown in (19). This modification overcomes the disadvantage of making decisions that are far-from-optimal with respect to downstream rewards due to solely focusing on the value of exploration. Nevertheless, if $v_t^{E,n} \ll \overline{V}_t^{x,n}$ we might explore only seldom and therefore converge to a "locally optimal" policy.

$$x_t^{n,\mathrm{E2}} = \underset{x_t^n \in \mathcal{X}_t^{\mathrm{R}}}{\arg\max} \left(\overline{V}_t^{x,n}\left(S_t^{x,n}\right) + v_t^{E,n}(S_t^n, K_t^n, x_t^n) \right) \tag{19}$$

Second, both the direct and the approximated downstream rewards can be added to the value of exploration when making a decision as proposed by Ryzhov et al. [33] and shown in (20). This modification ensures that towards the last iterations, when the value of exploration is approximately the same for many decisions, the exploitation decision is chosen. Exploitation in the last iterations will improve the downstream estimates if the policy learned is close to optimal. However, in this approach we need to be even more careful that the value of exploration is scaled properly, i.e., $v_t^{E,n} \sim \left(R_t\left(S_t^n, x_t\right) + \overline{V}_t^{x,n} \right)$, such that we actually do some exploration.

$$x_t^{n,\mathrm{E3}} = \underset{x_t^n \in \mathcal{X}_t^{\mathrm{R}}}{\arg\max} \left(R_t\left(S_t^n, x_t\right) + \overline{V}_t^{x,n}\left(S_t^{x,n}\right) + v_t^{E,n}(S_t^{x,n}, K_t^n, x_t^n) \right) \tag{20}$$

Third, we use the same rationale of the second modification but with a tighter control over the amount of exploration throughout the iterations. To achieve that in early iterations decisions are made according to traditional VPI

exploration, i.e., (13), and in later iterations follows pure exploitation, i.e., Line 5 in Algorithm 1, we introduce a weight $\alpha^n \in [0,1]$, as shown in (21). This iteration-dependent weight is close to one in early iterations and close to zero in later ones.

$$x_t^{n,\mathrm{E4}} = \underset{x_t^n \in \mathcal{X}_t^{\mathrm{R}}}{\arg\max} \left((1-\alpha^n) \left(R_t\left(S_t^n, x_t\right) + \overline{V}_t^{x,n}\left(S_t^{x,n}\right) \right) + \alpha^n v_t^{E,n}(S_t^{x,n}, K_t^n, x_t^n) \right) \tag{21}$$

The idea of exploring using the traditional VPI exploration decision in (13) is suitable as long as we are able to improve our approximation and it does not "cost" anything. However, some of the applications for which the idea was introduced have costs associated with exploration. This can happen, for instance, if the algorithm uses real-life observations of the exogenous information, or if the simulation of the exogenous information is so expensive that there is a limit on the number of iterations in ADP. This is the so-called "online" use of ADP [33], and for this case, the proposed modification in (20) seems reasonable. Although in our problem we use the "offline" version of ADP, which means we first learn the approximation without making real costs and then use the approximation to make decisions in real-life, we still suffer from exploration due to our finite horizon setting. Since we use backwards updating, exploration decisions at the final stages will affect the updates of the approximation at early stages. These effect would repeat over the iterations and hence the approximation that would be used after the ADP algorithm is finished would also suffer. This issue brings us to our last proposed modification: to be more conservative with updates resulting from exploration decisions.

5.2.3 The Update of the Approximation

The last modification we propose deals with the updates resulting from exploration decisions. Specifically, we propose adjusting the noise term $\sigma^{2,\mathrm{E}}$ in the updating equations (16) and (17). The general idea is that the higher this noise term is, the less the observed error (i.e., difference between the approximated downstream rewards and the observed ones) affects the approximations, since the difference can be partly attributed to "noise". Noise, in our context, has two causes: (i) fluctuations in the downstream rewards due to realizations of the random demand, and (ii) changes in the the policy (i.e., decisions made) due to the changing approximation $\overline{V}_t^n(S_t^{x,n})$. Typically, this noise term is assumed known and constant across all stages in an infinite horizon problem. In our problem, however, this would mean that $\sigma^{2,\mathrm{E}} = \eta^{\mathrm{E}}\ \forall t \in \mathcal{T}$, where η^{E} is the problem specific noise. This is not desirable as we explain below.

Suppose that we are at the same state and stage at iterations n and $n+1$ in the ADP algorithm. Although in both situations the feasible decision space is the same, decision $x_t^{n+1,*}$ might differ from decision $x_t^{n,*}$ because, from iteration n to iteration $n+1$, either the approximation $\overline{V}_t^n(S_t^{x,n})$ changed or the uncertainty knowledge K_t^n used to make exploration decisions changed. Since at earlier iterations changes in both the approximation and the uncertainty knowledge can be large due to their initialization, changes in the aforementioned decisions can be large. Furthermore, in a finite horizon problem with backward updates, changes in decisions at later stages would accumulate to earlier stages of the horizon, meaning that at early stages noise would be larger than at late

stages. To account for this decreasing nature of noise across the stages in the horizon, or across the iterations in our ADP algorithm, we propose three forms of modifying the noise term $\sigma^{2,\mathrm{E}}$.

First, we can let the noise term $\sigma_t^{2,\mathrm{E}}$ depend on the stage t as a linearly decreasing function of the constant noise term η^{E}, as shown in (22). The noise term η^{E} must, nevertheless, be tuned for the problem.

$$\sigma_t^{2,\mathrm{E2}} = \frac{T^{\max} - t}{T^{\max}} \eta^{\mathrm{E}} \tag{22}$$

Second, to deal with the noise due to changes in policy across the *iterations*, as well as stages, we can let the noise term depend on the uncertainty $\sigma_t^{2,n}(S_t^{x,n})$ about the impact of a decision, as shown in (23). The logic behind this modification is that, if we choose a decision that leads us to a highly uncertain post-decision state (i.e., high variance of the approximated downstream rewards of that post-decision state), then the resulting observation will have a lesser impact on our update. In VPI, the decision to visit a highly uncertain post-decision state is likely to be an exploration decision, and these decisions we typically want to contribute less since they can be far from optimal.

$$\sigma_t^{2,\mathrm{E3}}(S_t^{x,n}) = \sigma_t^{2,n}(S_t^{x,n}) \tag{23}$$

Third, we can combine the two previous ideas, as shown in (24). This is the most conservative of our proposals to modify the update of the approximation.

$$\sigma_{t,n}^{2,\mathrm{E4}}(S_t^{x,n}) = \frac{T^{\max} - t}{T^{\max}} \eta^{\mathrm{E}} + \sigma_t^{2,n}(S_t^{x,n}) \tag{24}$$

To recap, we proposed several modifications in three aspects of the exploration decisions to decrease the risk of negatively affecting our approximation. Overall, we have the following options to apply Bayesian exploration to our problem. First, we have two options to quantify the value of exploration: (15) and (18). Second, we have four options to make an exploration decision: (13), (19), (20), and (21). Third, we have four options to define the noise in the updating: constant, stage-dependent (22), iteration-dependent (23), and stage and iteration dependent (24). In the following section, we investigate which of the 2x4x4=32 combinations of modifications work best, and study the performance of ADP with Bayesian exploration compared to traditional ADP and other heuristics.

6 Numerical Experiments

In this section, we study the performance of our ADP designs through a series of numerical experiments. Specifically, we explore (i) the relation between the input parameters of our ADP designs and their performance in terms of learned and realized rewards, and (ii) the performance of using the learned policy from our ADP algorithms compared to a benchmark heuristic. In Section 6.1, we present our experimental design and setup. In Sections 6.2 and 6.3, we present and analyze the results of the two parts of our study. In Section 6.4, we finalize with a discussion on advantages and limitations of our ADP algorithms for scheduling freight in synchromodal networks.

6.1 Experimental Design

To test our ADP algorithms, we use three synchromodal network configurations and a planning horizon of $T^{\max} = 50$ days. These three networks, and their settings, are based on the intermodal consolidation network typologies suggested by [20], [44], and [5]. Each network increases in amount of consolidation opportunities (i.e., more services or terminals) compared to the previous one, as shown in Figures 2 to 4. In these figures, the axes denote the distance (in km) between the locations in the network, Q the capacity of each service in number of freights, and L^A the duration of each service in days. The duration of truck connections is one day. Remind that we assume an unlimited number of trucks, as motivated in Section 4.1. We consider transfer time not to be restrictive in any terminal in the network, i.e., $L^N_{i,t} = 0$, for all terminals i.

All networks span an area of 1000x500 km, and have the same location for origins and destinations. The distances between the origins and the destinations range from 800 to 1044 km, and the distances between an terminals close to the origin and terminals close to the destinations range from 500 to 854 km, to resemble distances that make consolidation for the long-haul desirable in Europe according to Woxenius [44]. We use a cost structure comparable to the one proposed by [3] and [15] to represent internal and external costs of intermodal and road freight transportation networks, and especially, to incorporate the differences in costs due to economies of scale of various transportation modes. For each day t in the horizon, the setup cost $B_{i,j,t}$ ranges between 169 and 425 for barges and trains and the variable cost $C_{i,j,d,t}$ ranges between 37 and 868 for barges, trains, and trucks. Further details of this cost structure, its cost-saving opportunities due to consolidation, and its decision challenges are given in Appendix B. For each day, the revenue $A_{i,j,d,t}$ is 868 for each freight picked up at its origin, independent of its origin or destination, and 0 otherwise. This entails that the entire revenue is received at the beginning of transportation (i.e., first mile), and afterwards only costs are incurred for each freight.

The number of freights that arrives at each origin, and their destination, varies according to the probability distributions shown in Appendix C. The time-window, upon arrival, is fixed to $r = 0$ and $k = 6$ for all freights (i.e., $p^{\mathrm{R}}_{0,i,t} = 1$ and $p^{\mathrm{K}}_{6,i,t} = 1$ for all origins i). This means that freights can immediately be transported, or postponed at most 2 days for a long-haul intermodal service to be feasible. Remind that, in synchromodality, there is no restriction on the transportation mode or number of transfers to use for a freight.

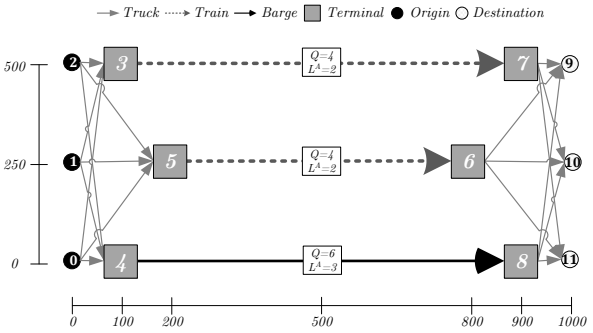

Figure 2: Network 1: point-to-point topology

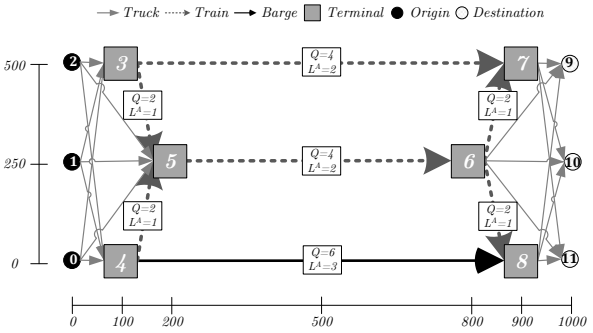

Figure 3: Network 2: collection-distribution topology

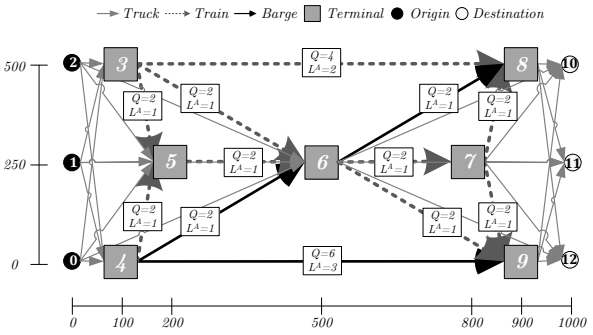

Figure 4: Network 3: hub-and-spoke topology

The initial state S_0 for each network contains six freights, one freight of each of the following characteristics: $F_{0,11(12),0,6,0}$, $F_{1,10(11),0,6,0}$, $F_{2,10(11),0,6,0}$, $F_{3,9(10),0,4,0}$, $F_{4,11(12),0,5,0}$, and $F_{5,11(12),0,1,0}$ (note that destinations in Network 3 are displayed between parenthesis). This initial state contains "average" freights on the origins plus a few freights in the network. Our choice of 50 days in the planning horizon ensures that the effect of the initial state in the rewards for the entire horizon is not so large by allowing enough variability (in

the arriving freights) to be present in the network. Naturally, the rewards for the horizon depend not only on the variability of the arriving freights but also on the decisions being made. For instance, postponing the transportation of freights will also postpone the moment that costs are incurred. In the end of the horizon, such decisions will be deemed good if there is no end-effect. For this reason, we include an end-effect (i.e., costs after day 50) by estimating the costs to send all freights remaining in the network using the benchmark heuristic, which is presented later on in Section 6.3.

To evaluate the performance of our ADP designs, we carry out two types of experiments: tuning and benchmark experiments. In the tuning experiments, we test several input parameters, such as the exploration probability ϵ in our ADP design with epsilon-greedy exploration and the noise term η^{E} in our ADP design with Bayesian exploration. Furthermore, we test the 32 possible VPI modifications. Our goal in these experiments is to provide insights into the relation of these parameters and their performance. We describe these experiments in detail and present their results in Section 6.2. In the benchmark experiments, we compare the performance of the best parameters of our two ADP designs resulting from the tuning experiments to a benchmark heuristic. Furthermore, we compare our best ADP design with the benchmark heuristic using various time-window settings. Our goal is to study the relation between time-window characteristics and the gains or losses of using ADP over using the benchmark heuristic. We describe these experiments and present their results in Section 6.3.

6.2 Tuning Experiments

In our tuning experiments, we study the two ADP designs from Section 5. We test the various modifications of our ADP design with Bayesian exploration and compare them to our ADP design with epsilon-greedy exploration. To make the comparison fair, we test several input values for the tunable parameters of each of the two designs, under the same conditions. Before presenting the results, we first describe the conditions under which we test our designs and the input values we use for the tunable parameters.

As described in Section 5.2, we are interested in the offline use of ADP, which is to first learn the approximation of downstream rewards within a simulated environment, and then use the approximation to make real-life decisions. For this reason, we measure performance in two ways: (i) learned approximation of downstream rewards after running the ADP algorithm, which we call *learned rewards*, and (ii) realized rewards of using the approximation for making decisions in a simulation, which we call *realized rewards*. The two of them are related to the MDP model's objective function in (6), the first relates to the optimal expected discounted rewards, and the second one relates to the policy that obtains these rewards. Although related, these two performance measurements are not necessarily the same. The basis function weights that our ADP algorithms learn can be far from the optimal rewards, but the resulting policy can be close to the optimal policy.

For each ADP design, we use $N = 50$ iterations and common random numbers, i.e., freights that arrive during each day of each iteration are the same across tested designs. Common random numbers are used to rule out differences in arrival processes of freights through the iterations of the ADP algorithms as

the source of differences in the learned and realized rewards among them. To test the resulting policy of each ADP design, we use 50 simulation runs of the planning horizon and, again, common random numbers which are different from the learning phase. Remind that the resulting policy, or function that maps each state to a decision, is given by the weights of the basis functions $[\theta_t^n]_{\forall t \in \mathcal{T}}$ and the restricted policies RP 1 (i.e., $\mathcal{X}_t^{\text{RP1}}$) and RP 2 (i.e., $\mathcal{X}_t^{\text{RP2}}$) defined in (8) and (9), respectively. These restricted policies significantly reduce the number of feasible decisions to evaluate. For example (see Appendix B), in a given state that has 2.6x10^8 feasible decisions, RP 1 reduces the number of decisions to evaluate to 1.9x10^4 and RP 2 reduces them to 5.8x10^4. These reductions, however, limit the performance that ADP can attain in learned and simulated rewards.

For the ADP design with ϵ-greedy exploration, we test three values of $\lambda = \{0.01, 0.1, 1\}$, which control the emphasis on recent observations during the updates in (12). We test four values of the probability of exploration $\epsilon = \{0, 0.3, 0.6, 0.9\}$. Remind that $\epsilon = 0$ means only exploitation decisions, while $\epsilon = 1$ means only exploration decisions. Furthermore, we test two ways of initializing the weights of the basis functions: (i) $\theta^0_{b,t} = 0$ for all characteristics $b \in \mathcal{B}$ and days $t \in \mathcal{T}$, and (ii) $\theta^0_{|\mathcal{B}|,t} = \beta\,(T^{\max} - t)/T^{\max}$ for the constant basis function and $\theta^0_{b,t} = 0$ for all other characteristics $b \in \mathcal{B}|b \neq |\mathcal{B}|$ and days $t \in \mathcal{T}$. The first initialization represents a case where we have no knowledge about the weights of the approximated rewards, and the second one represents a case where we have an estimate of the magnitude of the total downstream rewards β. In our case, β is defined as the rewards attained by the benchmark heuristic: $\beta = 38,036$ for Network 1, $\beta = 33,445$ for Network 2, and $\beta = 33,889$ for Network 3.

For the ADP design with Bayesian exploration, we initialize the basis function weights using the second option described before. We test four values of the noise term $\eta^{\text{E}} = \left\{10^2, 10^4, 10^6, 10^8\right\}$. For the initial covariance matrix, we test four values for the diagonal $\chi^{\text{C}} = \left\{10, 10^2, 10^3, 10^4\right\}$. We base our settings on Ryzhov et al. [33], who recommend that $\eta^{\text{E}} > \chi^{\text{C}}$ and that their ratio is of the order 10 or 10^2. For the weight α^n in modification (21), we test $\alpha^n = \{1/n, 10/(n+9), 100/(n+99)\}$. We test all the parameters described above for each of the 32 combinations of VPI modifications that we propose in Section 5.2.

Testing all tunable parameters and modifications of our two ADP designs for the three experimental networks results in more than 3500 experiments. Each experiment provides the learned and the realized rewards of ADP. Before discussing the details of the relation between ADP performance and the tunable parameters/modifications, we limit ourselves to present the results of the best parameters and modifications in Table 1, which consists of (i) the result for the tuned value of ϵ, λ, and β for the ADP design with ϵ-greedy exploration, and (ii) the tuned value of η^{E}, χ^{C}, and α^n, and the best combination of modifications in Section 5.2 for the ADP design with Bayesian exploration. We use the following acronyms within the tables and figures of this section: Restricted Policy (RP), Basis Functions (BF), and Value of Perfect Information (VPI).

Table 1: Maximum realized reward (i.e., tunned settings) and their corresponding learned reward for various ADP designs

ADP Design		Network 1		Network 2		Network 3	
		Realized Rewards	Learned Rewards	Realized Rewards	Learned Rewards	Realized Rewards	Learned Rewards
RP 1	BF	-7,994	38,219	-11,247	33,720	-16,548	-17,928
	BF + ϵ-greedy	-4,628	-6,984	-11,485	33,228	-18,172	-18,507
	BF + VPI	34,044	36,571	34,284	29,493	34,898	23,285
RP 2	BF	-4,912	-3,803	-11,734	34,060	-11,949	34,495
	BF + ϵ-greedy	880	37,386	-11,450	-12,091	-11,949	33,356
	BF + VPI	40,439	35,407	40,195	31,107	38,314	30,791

In Table 1, we observe the maximum realized rewards over all settings of the tunable parameters and VPI modifications, and their corresponding learned reward, for each ADP design. Three observations stand out. First, using VPI instead of the traditional ϵ-greedy strategy for exploration significantly improves the realized rewards. In fact, the policy resulting from the ϵ-greedy strategy, although better in most cases than the exploitation only (i.e., BF in the table), ends up in costs (i.e., negative rewards). Second, the difference in realized rewards between the two restricted policies varies per network and per design. Consider for instance Network 2, where the difference between the BF and BF + ϵ-greedy design across RP 1 and RP 2 is much smaller than the one from BF + VPI. Third, the accuracy of the approximation (i.e., difference between learned and realized rewards) varies per network and RP option for the traditional designs, but is more consistent for the BF+VPI design. BF+VPI underestimates the rewards by at most 11,000 whereas the traditional designs overestimate the rewards by at most 45,000. Although our focus is not on the learned rewards, these observations lead to discussion points, which we address in Section 6.4. We now focus on the relation between the tunable parameters/modifications of each ADP design and the realized rewards.

In the ADP design with basis functions and ϵ-greedy exploration, the tuned settings of most networks and RP options (all but one) ended up in negative rewards as seen in Table 1. All settings different from the tuned ones performed the same or worse. In practice, this means that the policy resulting from this ADP design would never be implemented. Therefore, we limit ourselves to two general observations only. First, there is no significant difference with respect to the way weights are initialized in each network and RP option. Second, there is at least one value of $\epsilon > 0$ that performs better than $\epsilon = 0$ for Network 1 and Network 2, but not in Network 3. Since this last observation hints that exploring pays off, but is not sufficient to fully escape the local-optima of exploitation decisions, we now focus on the analysis of our ADP + VPI design, which significantly outperforms the traditional design.

Before analyzing the VPI modifications, we analyze the tuning of the noise and Bayesian-belief related parameters. In Figure 5, we provide a comparison of different ratios $\eta^{\mathrm{E}}/\chi^{\mathrm{C}}$. In line with Ryzhov et al. [33], we observe that with ratios of 10^2 through 10^4, VPI works best on average over all modifications for both RPs. From this figure, we can gather two important insights for the ratio $\eta^{\mathrm{E}}/\chi^{\mathrm{C}}$. First, realized rewards and accuracy seem to improve with an increasing ratio until the best one (in our case 10^4). At the ratio with the best realized

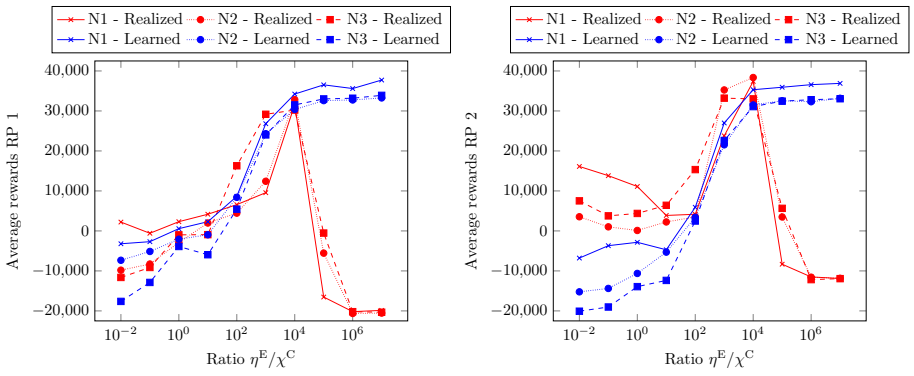

Figure 5: Comparison of average rewards (over all modifications) under different ratios η^{E}/χ^{C}

rewards we also find the smallest difference with the learned rewards. Second, at ratios larger than the "best-tuned" one, realized rewards rapidly decrease to the point of becoming costs rather than rewards, even though the learned rewards remain the same.

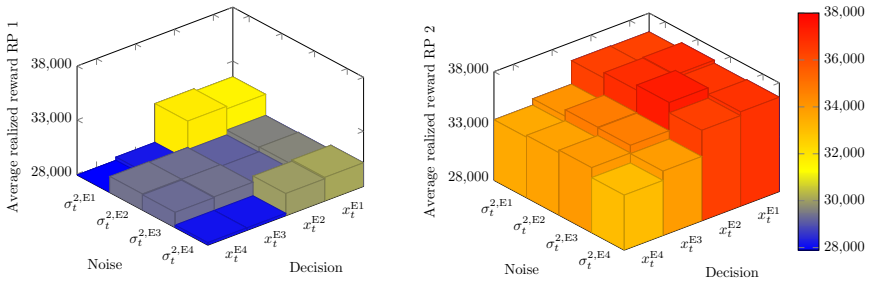

Figure 6: Comparison of average rewards (over all networks) for our proposed VPI modifications

At last, we analyze the performance of our VPI modifications. In Figure 6, we show the average realized rewards (over all networks) as a function of the decision and noise modifications explained in Sections 5.2.2 and 5.2.3, respectively. We exclude from our analysis our second proposed way of defining the value of exploration using the direct rewards since this performed significantly worse than the original definition of the value of exploration, for all decision and noise modifications. In Figure 6, the superscript E1 represents the original noise and decision definition in VPI, meaning that the upper corner corresponds to the original VPI design. We observe a significant difference between the average realized rewards for all modifications in RP 1 compared to RP 2, with all modifications in RP 2 performing better than the best modification in RP 1. This is to be expected since RP 2 has more decision freedom than RP 1. We also observe that, in RP 2, the modifications of how to make a decision have a larger impact than the modifications to the noise used when updating the approximation. Overall, including the downstream rewards in addition to the value of exploration when making decisions (i.e., x_t^{E2}) and letting the noise

depend only on the post-decision state of the given stage and iteration (i.e., σ_t^{E3}) are the best modifications tested for our problem. It seems that including the downstream rewards instead of the direct rewards helps the algorithm avoid the aforementioned greedy behavior and related worse performance. It also seems that considering the uncertainty of the post-decision state instead of a constant noise term when updating pays off. Using modifications x_t^{E2} and σ_t^{E3} for our BF+VPI design, we continue to our benchmark experiments.

6.3 Benchmark Experiments

In our benchmark experiments, we compare the realized rewards of our best ADP design (i.e., BF + VPI, with RP 2 and modifications x_t^{E2} and σ_t^{E3}) against a Benchmark Heuristic (BH). The objective is to compare the use of the learned ADP policy against a simpler but effective scheduling heuristic. The BH aims to use the intermodal services efficiently, i.e., consolidating as many freights as possible in a service once the setup costs for using that service can be covered. The BH consists of fours steps: (i) define the shortest and second shortest path for each freight to its final destination, considering only variable costs for services between terminals, (ii) calculate the savings between the shortest and second shortest path and define these as savings of the first intermodal service used in the shortest path, (iii) sort all freights in non-decreasing time-window length, i.e., closest due-day first, and (iv) for each freight in the sorted list, check whether the savings of the first intermodal service of its shortest path are larger than the setup cost for using this service; if so, use this service for the freight, if not, postpone the transportation of the freight. The pseudo-code for this heuristic can be found in Appendix E. Note that the BH is not subject to the restrictions imposed by RP 2, as ADP is.

Using the three networks from the previous section plus additional time-window distributions, we set up our experiments as follows. For each network, we replicate ten times the process of learning the ADP weights and simulating the use of the resulting policy. The use of the BH is also replicated ten times, using common random numbers with the corresponding ADP part, such that differences arise due to the scheduling differences and not the arriving freights. Identical to the tuning experiments of the previous section, one replication of the ADP process consists on running the ADP algorithm for 50 iterations and simulating the entire planning horizon 50 times. For the networks with different time-window distributions, we re-tune the noise parameters, since they are dependent on the inherent uncertainty of the problem. Here we fix the best ratio η^E/χ^C found, i.e., 10^4, but increase the values of η^E and χ^C.

Table 2: Average realized rewards (over the replications) of the BH and ADP

Scheduling policy	Network 1		Network 2		Network 3	
	Average	Gain	Average	Gain	Average	Gain
BH	37,347.36	-	33,066.79	-	32,963.79	-
ADP with RP 2	37,502.84	0%	36,867.96	11%	33,839.92	3%

Table 2 shows the average realized rewards, over the ten replications, using the same networks from the tuning experiments of the previous section. Remind that these networks have uncertainty in the amount of freight that arrives and

their destination, but not in their time-window. We observe that, for Network 1, ADP performs slightly better than the benchmark heuristic, but not with a significant difference. In Networks 2 and 3, however, ADP performs significantly better than the heuristic, with the difference between the two being the largest in Network 2. The differences in performance of both the BH and ADP across the networks seem to indicate two traits about scheduling freight in synchromodal transportation. First, the larger the complexity of the network is, the lower the average realized rewards are (remind that Network 1 is the simplest and 3 the most complex). Second, the gain of using ADP seems to be the largest in a more complex network (comparing Networks 1 and 2) up to a certain extent (comparing Networks 2 and 3). We come back to these traits in the following section.

In the experiments above, all freights that arrive are immediately released and have a time-window length of six days. With this time-window length, freights can be postponed at most 2 days for a long-haul intermodal service to be feasible. However, the length and uncertainty of the time-window of freights may affect the performance of an ADP algorithm [26]. To test this, we design three distributions for Release-Days (RD) and three distributions for Time-Window (TW) lengths, as shown in Tables 6 and 7 in Appendix C. Each distribution is categorized as short, medium, or long. Short RD means that 60% of the freights are released immediately while long RD means that 60% of the freights are released two days after arriving. Short TW means that 60% of the freights must be at their destination within 4 days after being released (i.e., cannot be postponed after released if a long-haul intermodal service is desired) while long RD means that 60% of the freights have a time-window length of 6 days. We follow the same procedure as before, with ten replications. Absolute results are shown in Table 8 (see Appendix D) and relative results are shown in Figure 7.

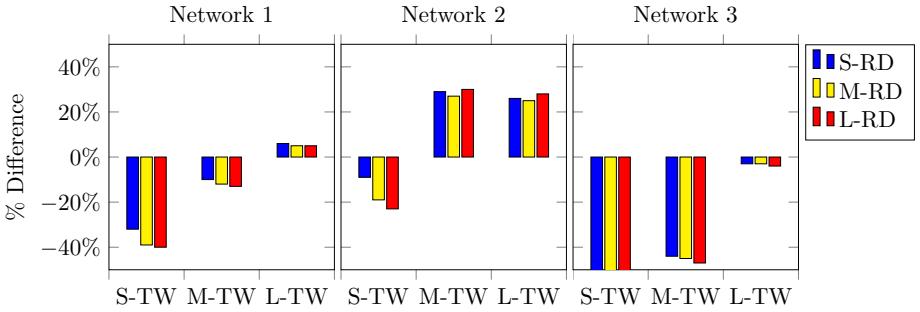

Figure 7: Percentage gain of ADP RP 2 over BH with respect to short (S), medium (M), and long (L) release days (RD) and time-window (TW) lengths.

As can be expected, the longer the distribution of time-window length, the larger the realized rewards are for both scheduling methods in each network, independent of the release-day. This happens because both methods are able to postpone the transportation of freight and anticipate on better consolidation opportunities. With respect to the performance of ADP, we observe that with short distributions of time-window lengths, the benchmark heuristic is better.

This is reasonable, since there are less options for postponing freights, and hence if a wrong postponement decision is made, the freight would have to be transported with an expensive alternative. Furthermore, it is reasonable that ADP performs worse than the BH since it is subject to the restrictions of RP 2, which enforces decisions on groups of freights rather than individual freights. With respect to RD distributions, we observe that ADP performs slightly better with short release-day distributions. Finally, Figure 7 seems to indicate that our ADP approach is useful in Network 2 but not in Networks 1 and 3. However, the weakness in our approach can be attributed mostly to the restricted decision space (RP 2) instead of to the learning of the basis function weights. In the following, we show the effect of RP 2 on our problem.

Suppose that the problem itself requires the two grouping restrictions of freights in RP 2: (i) all released freights that go to the same destination in each intermodal terminal must be transported together and (ii) all released freights in all origins that go to the same destination must be transported together. For ADP, these restrictions were already in place. For the BH, these restrictions mean that instead of doing the four steps for each freight, the four steps are done for each group of freights that fulfill the conditions of RP 2. We show the comparison between ADP and the BH with RP 2 in Table 3, Table 9 (see Appendix D), and Figure 8. The same setup for Table 2, Table 8 (see Appendix D), and Figure 7, is used, respectively.

Table 3: Average realized rewards (over the replications) considering RP 2

Scheduling policy	Network 1		Network 2		Network 3	
	Average	Gain	Average	Gain	Average	Gain
BH with RP 2	10,442.32	-	9,911.81	-	9,734.21	-
ADP with RP 2	37,502.84	259%	36,867.96	272%	33,839.92	248%

In Table 3, we observe that ADP significantly outperforms the BH with RP2 in all networks from the tuning experiments (i.e., no uncertainty in time-windows), with Network 2 resulting in the largest gain. With respect to the u uncertainty in time-windows, we observe that in each RD distribution, both scheduling policies achieve larger rewards with increasing TW distribution, although the differences are significantly larger for ADP. It seems that the longer the time-window is, the more ADP can look ahead for consolidation of freights compared to the BH with RP 2. Furthermore, although rewards across TW distributions differ among themselves, they look similar across RD distributions for each TW distribution (e.g., Medium TW distribution for Network 2 looks similar for Short, Medium, and Long RD). This makes sense since freights can be postponed only until their time-window allows it, independent of whether it is released immediately or in two days.

In Figure 8, we observe that our ADP design always outperforms the BH except for Network 3 with short time-window distribution. Actually, it is to be expected that ADP is less useful when there are short time-windows since postponement may directly result in using the expensive alternative mode (i.e., truck). In other words, anticipatory decisions do not make sense in such a problem. We also observe that for each TW distribution, the gains of ADP are larger in the case of short RD. There are two possible reasons for this: (i) the basis functions do not include any feature about release-days, so it is difficult

(impossible) to learn among post-decision freights with different release-days, (ii) when there are short release-days, rewards and costs are observed earlier, which improves the learning of the feature weights. These two reasons hint that the design of the approximation itself (i.e., the choice of basis functions) in transportation problems such as ours should take into account the time when rewards and costs are realized and their constraints.

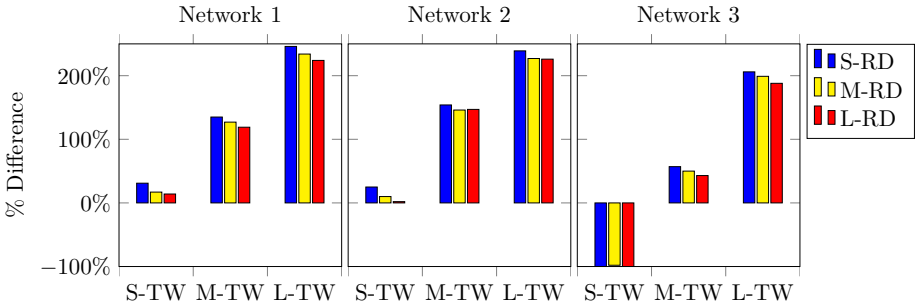

Figure 8: Percentage gain of ADP RP 2 over BH RP 2 with respect to short (S), medium (M), and long (L) release days (RD) and time-window (TW) lengths.

In general, our experiments show that including VPI in ADP improves its performance compared to traditional designs, and that this new design can lead to substantial gains over a benchmark heuristic for various problem settings. However, they also showed that a "one-size fits all" ADP solution is hard to achieve, and some tuning is necessary. In the following section, we reflect upon our results and discuss their implications.

6.4 Discussion

In our tuning experiments, we observed differences between the learned and realized rewards for most ADP designs. We observed that in the traditional ADP designs, a large learned reward did not necessarily results in a good policy. Most of the times, the learned rewards were positive while the realized rewards were negative. Although the dilemma of whether to focus on learning close-to-optimal rewards or a good policy relates to the issue of online or offline use of ADP, one can argue that both focuses are closely related and should, in theory, yield similar results. When looking for causes of the mismatch between learned and realized rewards, one can argue that problem characteristics such as the cost structure (i.e., revenue received at the beginning of transportation) and restricted decision space (i.e., some post-decision states are not attainable) have a strong influence on the mismatch. Nevertheless, we observe that through the inclusion of VPI, ADP is able to tackle these issues and significantly improve the results of traditional designs, both in learned and realized rewards. More specifically, we observed that during the learning phase, combining downstream rewards with the value of exploration in VPI was the best way of making exploration decisions, and that incorporating the uncertainty of a post-decision state when updating was the best way to update the approximation.

In our benchmark experiments, we observed that the restricted decision space is the weak point of our design. Compared to the benchmark heuristic

without restrictions, ADP achieve gains in only a few problem instances. Nevertheless, if we consider that the restricted decision space is part of the problem, ADP outperforms the benchmark heuristic drastically. These improved gains provide an indication of the gains that are possible if restrictions in the decision space of ADP would be removed. However, removing restrictions in the decision space can make it more computationally complex, which in turn can make exploration in the ADP learning phase more challenging. For example, we observed that the best rewards attained by ADP were in Network 1, which has the least complex decisions of all three networks. Naturally, exploration is more difficult in a settings with multi-period traveling times where multiple mode transfers are possible, such as in Networks 2 and 3, since the consequences of decisions span more than the next-period. This contrast between restricted and computationally complex decision space needs further research. There are at least three lines worth of consideration. First, restricting the policy during early iterations of ADP and then removing the restrictions in later iterations may overall result in a better policy learned with reasonable time. Second, using a heuristic decision policy rather than restrictions might also result in a better policy with less computational burden. Third, learning a good policy for a simple network (e.g., Network 1) and then using this feasible policy as starting approximation for a more complex network (e.g., Network 2) can also reduce the need for a large number of iterations.

7 Conclusions

In this paper, we developed an MDP model for the anticipatory scheduling of freight in a synchromodal transportation network and a heuristic solution based on ADP. We designed various ADP algorithms using the traditional constructs of basis functions and ϵ-greedy exploration, as well as methods from Bayesian exploration, specifically VPI. We described how the one-step look-ahead perspective of traditional ADP can make the algorithm flounder and end in a local-optimum, and how the ADP algorithm can escape this local-optimum and at the same time improve the solution by using the value of exploration from VPI. We proposed various modifications to VPI for infinite horizon problems to make it applicable to finite horizon ADP designs.

In a series of numerical experiments, we evaluated our ADP designs and our proposed modifications to VPI and provided insight into which modifications and tunable settings work best. We showed how VPI significantly improves the traditional ϵ-greedy strategy in a finite horizon problem, as long as exploring and updating in VPI is done slightly more conservative than in the original application of VPI in infinite horizon problems. We exemplified how ADP and VPI achieve significant gains in scheduling synchromodal freight transportation compared to a benchmark heuristic under different demand patterns. Finally, we reflected on the limitations of our study and possible ways to tackle these limitations. Further research about the reduction of the decision space, simplification of the network during the learning phase, and robustness of VPI settings in finite horizon problems is necessary for ADP to achieve the best performance in the scheduling of freight in a synchromodal transportation network considering demand uncertainty and performance over time.

Acknowledgment: This research has been partially funded by the Dutch Institute for Advanced Logistics, DINALOG, under the project SynchromodalIT.

References

[1] Ruibin Bai, Stein W. Wallace, Jingpeng Li, and Alain Yee-Loong Chong. Stochastic service network design with rerouting. *Transportation Research Part B: Methodological*, 60:50 – 65, 2014. ISSN 0191-2615. doi: http://dx.doi.org/10.1016/j.trb.2013.11.001. URL http://www.sciencedirect.com/science/article/pii/S0191261513001999.

[2] Behzad Behdani, Yun Fan, Bart Wiegmans, and Rob Zuidwijk. Multimodal schedule design for synchromodal freight transport systems. *European Journal of Transport & Infrastructure Research*, 16(3), 2016. URL https://d1rkab7tlqy5f1.cloudfront.net/TBM/Over%20faculteit/Afdelingen/Engineering%20Systems%20and%20Services/EJTIR/Back%20issues/16.3/2016_03_00%20Multimodal%20schedule%20design%20for%20synchromodal.pdf.

[3] Christian Bierwirth, Thomas Kirschstein, and Frank Meisel. On transport service selection in intermodal rail/road distribution networks. *Business Research*, 5(2):198–219, 2012. ISSN 2198-2627. doi: 10.1007/BF03342738. URL http://dx.doi.org/10.1007/BF03342738.

[4] Belgacem Bouzaiene-Ayari, Clark Cheng, Sourav Das, Ricardo Fiorillo, and Warren B. Powell. From single commodity to multiattribute models for locomotive optimization: A comparison of optimal integer programming and approximate dynamic programming. *Transportation Science*, 50(2):366–389, 2016. doi: 10.1287/trsc.2014.0536. URL https://doi.org/10.1287/trsc.2014.0536.

[5] An Caris, Cathy Macharis, and Gerrit K. Janssens. Decision support in intermodal transport: A new research agenda. *Computers in Industry*, 64(2):105 – 112, 2013. ISSN 0166-3615. doi: http://dx.doi.org/10.1016/j.compind.2012.12.001. URL http://www.sciencedirect.com/science/article/pii/S0166361512002047. Decision Support for Intermodal Transport.

[6] Anthony J. Craig, Edgar E. Blanco, and Yossi Sheffi. Estimating the {CO2} intensity of intermodal freight transportation. *Transportation Research Part D: Transport and Environment*, 22:49 – 53, 2013. ISSN 1361-9209. doi: http://dx.doi.org/10.1016/j.trd.2013.02.016. URL http://www.sciencedirect.com/science/article/pii/S1361920913000436.

[7] Teodor Gabriel Crainic, Mike Hewitt, and Walter Rei. Scenario grouping in a progressive hedging-based meta-heuristic for stochastic network design. *Computers & Operations Research*, 43:90 – 99, 2014. ISSN 0305-0548. doi: http://dx.doi.org/10.1016/j.cor.2013.08.020. URL http://www.sciencedirect.com/science/article/pii/S0305054813002268.

[8] Leonardo Campo Dall'Orto, Teodor Gabriel Crainic, Jose Eugenio Leal, and Warren B. Powell. The single-node dynamic service scheduling and

dispatching problem. *European Journal of Operational Research*, 170(1):1 – 23, 2006. ISSN 0377-2217. doi: http://dx.doi.org/10.1016/j.ejor.2004.06.016. URL http://www.sciencedirect.com/science/article/pii/S0377221704004552.

[9] Richard Dearden, Nir Friedman, and David Andre. Model based bayesian exploration. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, UAI'99, pages 150–159, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc. ISBN 1-55860-614-9. URL http://dl.acm.org/citation.cfm?id=2073796.2073814.

[10] Anny del Mar Agamez-Arias and José Moyano-Fuentes. Intermodal transport in freight distribution: a literature review. *Transport Reviews*, 0 (0):1–26, 2017. doi: 10.1080/01441647.2017.1297868. URL http://www.tandfonline.com/doi/abs/10.1080/01441647.2017.1297868.

[11] Mohammad Ghane-Ezabadi and Hector A. Vergara. Decomposition approach for integrated intermodal logistics network design. *Transportation Research Part E: Logistics and Transportation Review*, 89:53 – 69, 2016. ISSN 1366-5545. doi: http://dx.doi.org/10.1016/j.tre.2016.02.009. URL http://www.sciencedirect.com/science/article/pii/S1366554515300831.

[12] Gianpaolo Ghiani, Emanuele Manni, and Barrett W. Thomas. A comparison of anticipatory algorithms for the dynamic and stochastic traveling salesman problem. *Transportation Science*, 46(3):374–387, 2012. doi: 10.1287/trsc.1110.0374. URL http://dx.doi.org/10.1287/trsc.1110.0374.

[13] Gregory A. Godfrey and Warren B. Powell. An adaptive dynamic programming algorithm for dynamic fleet management, ii: Multiperiod travel times. *Transportation Science*, 36(1):40–54, 2002. doi: 10.1287/trsc.36.1.40.572. URL https://doi.org/10.1287/trsc.36.1.40.572.

[14] Justin C. Goodson, Barrett W. Thomas, and Jeffrey W. Ohlmann. Restocking-based rollout policies for the vehicle routing problem with stochastic demand and duration limits. *Transportation Science*, 50(2):591–607, 2016. doi: 10.1287/trsc.2015.0591. URL https://doi.org/10.1287/trsc.2015.0591.

[15] Milan Janic. Modelling the full costs of an intermodal and road freight transport network. *Transportation Research Part D: Transport and Environment*, 12(1):33 – 44, 2007. ISSN 1361-9209. doi: http://dx.doi.org/10.1016/j.trd.2006.10.004. URL http://www.sciencedirect.com/science/article/pii/S1361920906000794.

[16] Behzad Kordnejad. Intermodal transport cost model and intermodal distribution in urban freight. *Procedia - Social and Behavioral Sciences*, 125:358 – 372, 2014. ISSN 1877-0428. doi: http://dx.doi.org/10.1016/j.sbspro.2014.01.1480. URL http://www.sciencedirect.com/science/article/pii/S1877042814015183.

[17] Le Li, Rudy R. Negenborn, and Bart De Schutter. Intermodal freight transport planning a receding horizon control approach. *Transportation Research Part C: Emerging Technologies*, 60:77 – 95, 2015. ISSN 0968-090X. doi: http://dx.doi.org/10.1016/j.trc.2015.08.002. URL http://www.sciencedirect.com/science/article/pii/S0968090X15002685.

[18] Arnt-Gunnar Lium, Teodor Gabriel Crainic, and Stein W. Wallace. A study of demand stochasticity in service network design. *Transportation Science*, 43(2):144–157, 2009. doi: 10.1287/trsc.1090.0265. URL http://dx.doi.org/10.1287/trsc.1090.0265.

[19] Hong K. Lo, Kun An, and Wei hua Lin. Ferry service network design under demand uncertainty. *Transportation Research Part E: Logistics and Transportation Review*, 59:48 – 70, 2013. ISSN 1366-5545. doi: http://dx.doi.org/10.1016/j.tre.2013.08.004. URL http://www.sciencedirect.com/science/article/pii/S1366554513001440.

[20] C Macharis and Y.M Bontekoning. Opportunities for {OR} in intermodal freight transport research: A review. *European Journal of Operational Research*, 153(2):400 – 416, 2004. ISSN 0377-2217. doi: http://dx.doi.org/10.1016/S0377-2217(03)00161-9. URL http://www.sciencedirect.com/science/article/pii/S0377221703001619. Management of the Future MCDA: Dynamic and Ethical Contributions.

[21] William G Macready and David H Wolpert. Bandit problems and the exploration/exploitation tradeoff. *IEEE Transactions on evolutionary computation*, 2(1):2–22, 1998. URL http://ieeexplore.ieee.org/document/728210/#.

[22] Martijn R. K. Mes and Maria-Eugenia Iacob. Synchromodal transport planning at a logistics service provider. In Henk Zijm, Matthias Klumpp, Uwe Clausen, and ten Michael Hompel, editors, *Logistics and Supply Chain Innovation: Bridging the Gap between Theory and Practice*, chapter Synchromodal Transport Planning at a Logistics Service Provider, pages 23–36. Springer International Publishing, Cham, 2016. ISBN 978-3-319-22288-2. doi: 10.1007/978-3-319-22288-2_2. URL http://dx.doi.org/10.1007/978-3-319-22288-2_2.

[23] J.L. Nabais, R.R. Negenborn, R.B. Carmona Bentez, and M. Ayala Botto. Achieving transport modal split targets at intermodal freight hubs using a model predictive approach. *Transportation Research Part C: Emerging Technologies*, 60:278 – 297, 2015. ISSN 0968-090X. doi: http://dx.doi.org/10.1016/j.trc.2015.09.001. URL http://www.sciencedirect.com/science/article/pii/S0968090X15003241.

[24] Arturo Pérez Rivera and Martijn Mes. Dynamic multi-period freight consolidation. In Francesco Corman, Stefan Voß, and Rudy R. Negenborn, editors, *Computational Logistics: 6th International Conference, ICCL 2015, Delft, The Netherlands, September 23-25, 2015, Proceedings*, volume 9335 of *Lecture Notes in Computer Science*, pages 370–385. Springer International Publishing, 2015. ISBN 978-3-319-24263-7. doi: 10.1007/978-3-319-24264-4_26. URL http://dx.doi.org/10.1007/978-3-319-24264-4_26.

[25] Arturo Pérez Rivera and Martijn Mes. Service and transfer selection for freights in a synchromodal network. In Ana Paias, Mario Ruthmair, and Stefan Voß, editors, *Computational Logistics: 7th International Conference, ICCL 2016, Lisbon, Portugal, September 7-9, 2016, Proceedings*, pages 227–242. Springer International Publishing, Cham, 2016. ISBN 978-3-319-44896-1. doi: 10.1007/978-3-319-44896-1_15. URL `http://dx.doi.org/10.1007/978-3-319-44896-1_15`.

[26] Arturo Pérez Rivera and Martijn Mes. Anticipatory freight selection in intermodal long-haul round-trips. *Transportation Research Part E: Logistics and Transportation Review*, 105(Supplement C):176 – 194, 2017. ISSN 1366-5545. doi: https://doi.org/10.1016/j.tre.2016.09.002. URL `http://www.sciencedirect.com/science/article/pii/S1366554515303392`.

[27] Warren B. Powell. *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. John Wiley & Sons Inc., second edition edition, 2011.

[28] Warren B. Powell and Ilya O. Ryzhov. *Optimal Learning*. John Wiley & Sons, Inc., 2012. ISBN 9781118309858. doi: 10.1002/9781118309858. URL `http://dx.doi.org/10.1002/9781118309858`.

[29] Warren B. Powell, Hugo P. Simao, and Belgacem Bouzaiene-Ayari. Approximate dynamic programming in transportation and logistics: a unified framework. *EURO Journal on Transportation and Logistics*, 1(3):237–284, Sep 2012. ISSN 2192-4384. doi: 10.1007/s13676-012-0015-8. URL `https://doi.org/10.1007/s13676-012-0015-8`.

[30] Bart Riessen, Rudy R. Negenborn, and Rommert Dekker. Synchromodal container transportation: An overview of current topics and research opportunities. In Francesco Corman, Stefan Voß, and R. Rudy Negenborn, editors, *Computational Logistics: 6th International Conference, ICCL 2015, Delft, The Netherlands, September 23-25, 2015, Proceedings*, Lecture Notes in Computer Science, chapter Synchromodal Container Transportation: An Overview of Current Topics and Research Opportunities, pages 386–397. Springer International Publishing, Cham, 2015. ISBN 978-3-319-24264-4. doi: 10.1007/978-3-319-24264-4_27. URL `http://dx.doi.org/10.1007/978-3-319-24264-4_27`.

[31] Bart Van Riessen, Rudy R Negenborn, Rommert Dekker, and Gabriel Lodewijks. Service network design for an intermodal container network with flexible transit times and the possibility of using subcontracted transport. *International Journal of Shipping and Transport Logistics*, 7(4):457–478, 2015. doi: http://dx.doi.org/10.1504/IJSTL.2015.069683. URL `http://www.inderscienceonline.com/doi/abs/10.1504/IJSTL.2015.069683`.

[32] Ilya O. Ryzhov and Warren B. Powell. Bayesian active learning with basis functions. In *IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*, pages 143–150, 2011. URL `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5967365`. Winner, Best Paper, IEEE International Symposium on Adaptive Dynamic Programming and Reinforcement Learning, 2011.

[33] Ilya O. Ryzhov, Martijn R.K. Mes, Warren B. Powell, and Gerald A. van den Berg. Bayesian exploration for approximate dynamic programming. *Operations Research*, Submitted, 2017. URL http://scholar.rhsmith.umd.edu/iryzhov/publications/bayesian-exploration-approximate-dynamic-programming?destination=node/1319.

[34] Hugo P. Simao, Jeff Day, Abraham P. George, Ted Gifford, John Nienow, and Warren B. Powell. An approximate dynamic programming algorithm for large-scale fleet management: A case application. *Transportation Science*, 43(2):178–197, 2009. doi: 10.1287/trsc.1080.0238. URL http://dx.doi.org/10.1287/trsc.1080.0238.

[35] M. SteadieSeifi, N.P. Dellaert, W. Nuijten, T. Van Woensel, and R. Raoufi. Multimodal freight transportation planning: A literature review. *European Journal of Operational Research*, 233(1):1 – 15, 2014. ISSN 0377-2217. doi: http://dx.doi.org/10.1016/j.ejor.2013.06.055. URL http://www.sciencedirect.com/science/article/pii/S0377221713005638.

[36] Malcolm Strens. A bayesian framework for reinforcement learning. In *ICML*, pages 943–950, 2000. URL http://web.eecs.utk.edu/~itamar/courses/ECE-692/paper1c.pdf.

[37] Richard S. Sutton and Andrew G. Barto. *Introduction to Reinforcement Learning*. MIT Press, Cambridge, MA, USA, 1st edition, 1998. ISBN 0262193981. URL https://mitpress.mit.edu/books/reinforcement-learning.

[38] Richard S. Sutton, Hamid Reza Maei, Doina Precup, Shalabh Bhatnagar, David Silver, Csaba Szepesvári, and Eric Wiewiora. Fast gradient-descent methods for temporal-difference learning with linear function approximation. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 993–1000, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-516-1. doi: 10.1145/1553374.1553501. URL http://doi.acm.org/10.1145/1553374.1553501.

[39] Marlin W. Ulmer, Dirk C. Mattfeld, and Felix Köster. Budgeting time for dynamic vehicle routing with stochastic customer requests. *Transportation Science*, Articles in advance, 2017. doi: 10.1287/trsc.2016.0719. URL https://doi.org/10.1287/trsc.2016.0719.

[40] W. J. A. van Heeswijk, M. R. K. Mes, and J. M. J. Schutten. The delivery dispatching problem with time windows for urban consolidation centers. *Transportation Science*, Articles in advance, 2017. doi: 10.1287/trsc.2017.0773. URL https://doi.org/10.1287/trsc.2017.0773.

[41] Matej Črepinšek, Shih-Hsi Liu, and Marjan Mernik. Exploration and exploitation in evolutionary algorithms: A survey. *ACM Comput. Surv.*, 45 (3):35:1–35:33, July 2013. ISSN 0360-0300. doi: 10.1145/2480741.2480752. URL http://doi.acm.org/10.1145/2480741.2480752.

[42] Joannès Vermorel and Mehryar Mohri. Multi-armed bandit algorithms and empirical evaluation. In João Gama, Rui Camacho, Pavel B. Brazdil,

Alípio Mário Jorge, and Luís Torgo, editors, *Machine Learning: ECML 2005: 16th European Conference on Machine Learning, Porto, Portugal, October 3-7, 2005. Proceedings*, pages 437–448. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005. ISBN 978-3-540-31692-3. doi: 10.1007/11564096_42. URL https://doi.org/10.1007/11564096_42.

[43] Nicole Wieberneit. Service network design for freight transportation: a review. *OR Spectrum*, 30(1):77–112, 2008. ISSN 0171-6468. doi: 10.1007/s00291-007-0079-2. URL http://dx.doi.org/10.1007/s00291-007-0079-2.

[44] Johan Woxenius. Generic framework for transport network designs: Applications and treatment in intermodal freight transport literature. *Transport Reviews*, 27(6):733–749, 2007. doi: 10.1080/01441640701358796. URL http://www.tandfonline.com/doi/abs/10.1080/01441640701358796.

[45] M. Zhang and A.J. Pel. Synchromodal hinterland freight transport: Model study for the port of rotterdam. *Journal of Transport Geography*, 52:1 – 10, 2016. ISSN 0966-6923. doi: http://dx.doi.org/10.1016/j.jtrangeo.2016.02.007. URL http://www.sciencedirect.com/science/article/pii/S096669231600020X.

A Modeling Assumptions

To simplify the formulation of the MDP model in Section 4, we make several assumptions and enforce certain conditions. These assumptions and conditions apply to our model, but not necessarily to our problem. In this Appendix, we briefly describe the possible modifications to the MDP model such that each assumption or condition can be bypassed.

The first condition we impose in our model is the separation between origin, intermodal, and destination nodes. If an intermodal terminal is also an origin or destination of freights, a duplicate node can be included in the set of origins or destinations and their related parameters can be changed accordingly. The second assumption in our model relates to the unlimited capacity for the first and third type of arcs, i.e., services beginning at an origin or ending at a destination. In other words, we assume that the pre- and end-haulage operations of our synchromodal network are not restrictive. If there is a restriction, this must be added to the decision space and, in case of penalization or loss of freight, the transition function must be changed accordingly.

One of the modeling challenges we avoid is having more than one service between two terminals. We mentioned before that duplicate intermodal nodes can be added, and that services need to be modified accordingly. However, this modification is more than just altering the arcs, it involves: (i) modifying the transition function such that the same freights appear/disappear in the duplicated nodes and (ii) modifying the decision space constraints such that service capacities going to/from the duplicated nodes are respected and such that no more than the existing freights (i.e., not duplicated) can be transported.

B Experimental Cost Structure, Consolidation Opportunities, and Decision Challenges

To model the costs and the effects of freight consolidation (i.e., setup costs), we use the costs per km presented in [3] and the model with which they were calculated originally in [15]. This cost model was developed for internal and external costs of intermodal and road freight transportation networks, and the logic behind it is that costs decrease non-linearly with distance and at different rates depending on the mode. The variable cost (i.e., euro per km) for truck is $5.46d^{-0.278}$, for train is $0.58d^{-0.26}$, and for barge is $0.46d^{-0.26}$, where d is the distance between two locations. For the fixed or setup cost (i.e., euro per service independent of the number of freight) of a service for the train is $q((1560^{0.74})/q+40)$ and for barge is $0.8q((1560^{0.74})/q+40)$, where q is the capacity of the service. We refer the reader to [15] for a thorough explanation on the cost model. In the next paragraph, we describe the consolidation opportunities and challenges of making decisions in each network.

Network 1 represents the so-called point-to-point topology [20]. Although there are no transfers in this network, there are three consolidation opportunities for each origin, namely the two train and barge services. The complexity of the decisions in this network is two-fold: (i) the restrictions imposed by the capacity of each service and (ii) the relation between the transportation duration with the time-window of freights. Network 2 represents the so-called collection-distribution topology [20]. In this network, there are four additional services, and new transfers connected to the central terminals, compared to Network 1. The new consolidation opportunities bring two additional challenges to those of Network 1, the trade-off of using truck against truck-and-train to (i) bring a freight from its origin to the start of the long-haul and (ii) bring a freight from the end of the long-haul to its destination. Network 3 represents the so-called hub-and-spoke network topology [20]. In this network, there is one additional terminal, four additional services, and new transfers connected to the new terminal, compared to Network 2. The number of paths a freight can take from its origin to its destination significantly increases, and thus the complexity of the decisions increases as well. To exemplify this complexity, we compare the number of decisions in the feasible decision space of Network 3 in the following paragraph.

In Network 3, freight from any origin, independent of the destination, can be transported to any of the closest four terminals, transported to its destination, or postponed. Now, if we would have one released freight with the maximum time-window in each origin, for each destination, we would have $3 \times 3 \times (4 + 1 + 1) = 54$ possible decisions for freights at the origins. Now, consider we have similar freights in Terminals 3, 4, and 6, which all have three services, and we will have 45 possible decisions for freights at those terminals, and $54 \times 45 = 2430$ decisions for freight at origins and Terminals 3, 4, and 6. Following a similar logic, we would end-up with 4.3×10^5 possible decisions in the network if at each terminal there would be one released freight with a large enough time-window (i.e., for services to be feasible) for each destination. If there would be two freights instead of one, we would end up with 2.6×10^8 possible decisions. In contrast, the same two freights with a large enough time-window for each destination, at each location, would end up in 1.9×10^4 in the

restricted policy 1 and 5.8x10^4 in the restricted policy 2 defined in (8) and (9), respectively.

C Experimental Probability Distributions

Table 4: Freight probability distributions for all networks

Origin 0		Origin 1		Origin 2	
Freights f	Prob. $p^{\mathrm{F}}_{f,0,t}$	Freights f	Prob. $p^{\mathrm{F}}_{f,1,t}$	Freights f	Prob. $p^{\mathrm{F}}_{f,2,t}$
0	0.14	0	0.22	0	0.37
1	0.27	1	0.33	1	0.37
2	0.27	2	0.25	2	0.18
3	0.18	3	0.13	3	0.06
4	0.14	4	0.07	4	0.02

Table 5: Destination probability distributions for all networks

Origin 0		Origin 1		Origin 2	
Destination* d	Prob. $p^{\mathrm{D}}_{d,0,t}$	Destination* d	Prob. $p^{\mathrm{D}}_{d,1,t}$	Destination* d	Prob. $p^{\mathrm{D}}_{d,2,t}$
9 (10)	0.1	9 (10)	0.33	9 (10)	0.14
10 (11)	0.1	10 (11)	0.34	10 (11)	0.29
11 (12)	0.8	11 (12)	0.33	11 (12)	0.57

*Destinations for Network 3 are displayed between parenthesis.

Table 6: Release Day (RD) distributions for all networks in the Benchmark Experiments

Short		Medium		Long	
RD	Prob.	RD	Prob.	RD	Prob.
0	0.6	0	0.33	0	0.1
1	0.3	1	0.34	1	0.3
2	0.1	2	0.33	2	0.6

Table 7: Time-window (TW) length distributions for all networks in the Benchmark Experiments

Short		Medium		Long	
TW	Prob.	TW	Prob.	TW	Prob.
4	0.6	4	0.33	4	0.1
5	0.3	5	0.34	5	0.3
6	0.1	6	0.33	6	0.6

D Absolute Results of the Benchmark Experiments

Table 8: Average realized rewards for different time-window distributions

RD distribution	TW distribution	Network 1		Network 2		Network 3	
		BH	ADP	BH	ADP	BH	ADP
Short	Short	17,862	12,131	12,339	11,289	22,191	(19)
	Medium	25,286	22,775	18,232	23,486	26,634	15,001
	Long	33,007	35,111	25,805	32,524	30,680	29,745
Medium	Short	17,812	10,938	12,160	9,877	22,141	209
	Medium	25,302	22,267	18,052	23,015	26,573	14,612
	Long	32,805	34,508	25,420	31,806	30,473	29,502
Long	Short	17,773	10,724	12,062	9,281	22,256	(44)
	Medium	25,276	21,956	17,951	23,422	26,568	14,167
	Long	32,876	34,511	25,401	32,462	30,467	29,274

Table 9: Average realized rewards for different time-window distributions considering RP 2

RD distribution	TW distribution	Network 1		Network 2		Network 3	
		BH RP 2	ADP RP 2	BH RP 2	ADP RP 2	BH RP 2	ADP RP 2
Short	Short	9,273	12,131	9,014	11,289	9,374	(19)
	Medium	9,677	22,775	9,244	23,486	9,537	15,001
	Long	10,151	35,111	9,601	32,524	9,728	29,745
Medium	Short	9,322	10,938	9,003	9,877	9,494	209
	Medium	9,814	22,267	9,338	23,015	9,728	14,612
	Long	10,341	34,508	9,719	31,806	9,881	29,502
Long	Short	9,438	10,724	9,074	9,281	9,601	(44)
	Medium	10,037	21,956	9,485	23,422	9,890	14,167
	Long	10,643	34,511	9,944	32,462	10,150	29,274

E Benchmark Heuristic

Algorithm 3 Benchmark heuristic

Define *List* := released freights (i.e., $F_{i,d,0,k,t}$) sorted by (a) non-decreasing time-window k and (b) non-increasing size $F_{i,d,0,k,t}$
Randomize order of freights with the same time-window and size in *List*
for $F_{i,d,0,k,t}$ **in** *List* **do**
  if $t > L^A_{i,d,t}$ (i.e., freight is *not* urgent) **then**
    Define $P^1_{i,d,k}$:= path to transport $F_{i,d,0,k,t}$ to its destination with the cheapest variable cost
    Define $C^1_{i,d,k}$:= variable cost of $P^1_{i,d,k}$
    Define $P^2_{i,d,k}$:= path to transport $F_{i,d,0,k,t}$ to its destination with the second cheapest variable cost
    Define $C^2_{i,d,k}$:= variable cost of $P^2_{i,d,k}$
    Define $S_{i,d,k}$:= Savings between the two cheapest-variable-cost paths of f (i.e., $C^2_{i,d,k} - C^1_{i,d,k}$)
  else
    Schedule $F_{i,d,0,k,t}$ in truck
  end if
end for
Initialize $T(i,j) := 0$ for all services $(i,j) \in \mathcal{A}_t$
for $F_{i,d,0,k,t}$ **in** *List* **do**
  Define i := first element of $P^1_{i,d,k}$ (i.e., location o $F_{i,d,0,k,t}$)
  Define j := second element of $P^1_{i,d,k}$ (i.e., next location to transport $F_{i,d,0,k,t}$)
  Increase $T(i,j) :+ S_f$ (i.e., adding savings from a freight to overall savings of service (i,j))
end for
for $F_{i,d,0,k,t}$ **in** *List* **do**
  if $t > L^A_{i,d,t}$ (i.e., freight is *not* urgent) **then**
    Define i := first element of $P^1_{i,d,k}$ (i.e., location o $F_{i,d,0,k,t}$)
    Define j := second element of $P^1_{i,d,k}$ (i.e., next location to transport $F_{i,d,0,k,t}$)
    if $T(i,j) >= B_{i,j,t}$ **then**
      Schedule $F_{i,d,0,k,t}$ in service (i,j) if there is available capacity.
    end if
  end if
end for