Transportation Research Part C 181 (2025) 105366

Contents lists available at ScienceDirect

Transportation Research Part C

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

# Resilient synchromodal transport through learning assisted hybrid simulation optimization model

Satrya Dewantara [a], Siyavash Filom [b], Saiedeh Razavi [b], Bilge Atasoy [c], Yimeng Zhang [a,d], Mahnam Saeednia [a,*]

[a] Department of Transport and Planning, Faculty of Civil Engineering, TU Delft, Delft, the Netherlands
[b] Civil Engineering, McMaster University, Hamilton, Canada
[c] Department of Maritime and Transport Technology, Faculty of Mechanical Engineering, TU Delft, Delft, the Netherlands
[d] School of Transportation & Logistics, Southwest Jiaotong University, Chengdu, China

ARTICLE INFO

*Keywords:*
Synchromodality
Resilient freight transport
Learning-based decision support framework
Disruptions
Discrete-event simulation
Reinforcement learning
Simulation-optimization

ABSTRACT

Disruptions and uncertainties can significantly reduce the efficiency of conventional intermodal transport, often leading to severe economic losses and deterioration in service levels. To mitigate the negative impacts of disruptions on the shipments, our research leverages the flexibility of synchromodality and develops a learning-based modular framework for disruption management. By utilizing a hybrid simulation-optimization modeling approach, the framework effectively captures disruptions and generates dynamic response strategies. Through the integration of Reinforcement Learning (RL), the proposed approach re-plans under disruptions, accounting for their stochastic characteristics, enabling swift and effective decision-making in real-time scenarios. Results are compared against two policies, always wait and always reassign, highlighting the superior performance of the RL approach, when exposed to a certain disruption profile, with comparable or better decisions compared to other policies in response to disruptions. Additionally, results are compared against a benchmark policy to test an alternative reward mechanism, demonstrating that integrating a cost-based reward mechanism increases its resilience and results in lower costs, especially in the case of more frequent and low to moderately severe disruptions.

## 1. Introduction

Disruptions and uncertainties negatively contribute to the efficiency of conventional intermodal transport (Delbart et al., 2021) resulting in severe economic losses and an undesired level of service. Disruptions with low occurrence probability but high impact such as COVID-19 increased logistic costs by 12 % globally (Rodríguez-Clare et al., 2023). From another spectrum where disruptions occur frequently, a study by Schlake et al. (2011) estimate $15.2 Million loss in a year due to train delays. This estimate made in 2011 will have a significantly higher value today. The concept of synchromodality introduces flexibility to intermodal transport, leading to an increase in its attractiveness by offering shippers a wider range of combined routes. This flexibility provides added value by improving the trade-off between cost and time (Tavasszy et al., 2015). The inherent flexibility in synchromodality systems enables them to respond more effectively to disruptions. For instance, in theory, containers initially planned for transport by barge can be seamlessly rerouted to trains in the event of a disruption in the barge network. This paper explores how this flexibility can be harnessed to create resilient intermodal transport systems.

* Corresponding author.
*E-mail address:* m.saeednia@tudelft.nl (M. Saeednia).

https://doi.org/10.1016/j.trc.2025.105366
Received 26 December 2024; Received in revised form 18 June 2025; Accepted 28 September 2025
Available online 15 October 2025

The resilience of a freight transport network is defined by its ability to recover from disruptions and is measured by the effort required to restore normal operations (Chen and Miller-Hooks, 2012). Several studies have proposed a resilient synchromodal transport by exploiting its flexibility feature. For instance, Qu et al. (2019) introduce a re-planning model to re-route shipments in response to disruptions. Hrušovský et al. (2021) combine agent-based and discrete-event simulations to monitor transport activities and integrate optimization to execute offline and online planning.

However, these approaches assume a-priori information about disruptions-an assumption often unrealistic in practice (Wide et al., 2022). They also suffer from computational challenges due to high dimensionality, complicating their application in large-scale or dynamic environments (Jaimungal, 2022). In contrast, RL operates online without requiring initial disruption knowledge. Crucially, RL can adapt to evolving disruption types over time, whereas stochastic or robust optimization methods rely solely on predefined disruption profiles. Furthermore, RL policies can learn terminal-specific strategies within synchromodal transport networks, while stochastic and robust optimization are constrained to uniform disruption responses.

To address this challenge, Zhang et al. (2023) integrate RL into a replanning model and simulates the synchromodal transport using a rolling horizon approach with an event-triggered mechanism to simulate synchromodal transport dynamics. Since the RL technique relies heavily on its interaction with the environment (Memarian and Doleck, 2024), creating an environment that mimics real-world dynamics can improve the outcome for a decision support system. In light of this, this paper presents a Decision Support System (DSS) for managing disruptions by leveraging the increased flexibility offered by synchromodality with contributions as follows:

- The proposed DSS adopts a learning-based approach to support decision-makers in real-time and employs a modular architecture for extensibility;
- It integrates a discrete-event simulator and a comprehensive disruption profile for enhanced capture of the dynamic nature of disruptions in real-world operations;
- The framework adjusts plans based on actual costs incurred by decisions, considering tradeoffs among different cost components while accounting for shipment volume and delay length.
- It adds to the prior research by introducing a cost-based reward mechanism and providing practical deployment thresholds by comparison.

Developing such a comprehensive model presents challenges but promises practical solutions for industry implementation, shifting freight transport paradigms towards sustainability and flexibility. The modular framework, integrating optimization, simulation, and machine learning, facilitates plug-and-play connections with existing or under-development models, making it adaptable across diverse port-inland ecosystems and expanding the solution space for disruption management.

The paper is organized as follows: Section 2 provides a brief literature review focusing on resilient synchromodal frameworks. Section 3 outlines the problem description, while Section 4 details the methodology and framework modeling. The model is implemented in Section 5 using a network adapted from a real-world operation. Finally, the research is concluded in Section 6 to answer the main research question and provide directions for possible future research.

## 2. Literature review

This section investigtes the current state of literature that consider different aspects of disruption management in synchromodal transport. It focuses on exploring dynamic models of synchromodality through various approaches, including the use of RL to mitigate disruptions.

Dynamic models are critical to realize synchromodal transport and have been proposed in several studies. Qu et al. (2019) develop a Synchromodal Transportation Re-planning (STP) for hinterland transport using a mixed integer linear programming (MILP). Using a different approach, Guo et al. (2020) propose a dynamic matching problem to deal with uncertain shipment requests. In this model, the shipment requests are not completely known, but rather sequentially announced using a rolling horizon approach. This approach is adopted by Guo et al. (2022) for a global shipment matching problem where it is extended by incorporating disruptions in the service network. The reaction to disruption in these two models is through the replanning of the containers.

Other studies propose agent-based models to compare the performance of unimodal, intermodal, and synchromodal for cost, time, and emissions. Ambra et al. (2019) apply a synchromodal scenario where each agent has a decision logic to reroute to the nearest and cheapest terminal if there is a disruption in the network and monitor the impact on cost, time, and emission. Di Febbraro et al. (2016) propose an agent-based framework for cooperative planning based on decentralized optimization with a negotiation scheme. It breaks down the problem into several sub-problems and lets the agents communicate with other agents to achieve each objective under disrupted scenarios. The model provides a sequence plan and re-plans it when exogenous events occur. A decision support system is proposed by Hrušovský et al. (2021) using a hybrid simulation-optimization model under a synchromodal framework. The study combines an agent-based and discrete-event simulation to monitor transport activities. It employs an offline model to create the initial plan and an online model to react to the disruptions and selects one of three possible policies: wait, transshipment, or detour. The disruptions are categorized according to frequency and duration by assigning them to a random variable in the simulation. The online model will be triggered if a disruption occurs. The result of the study shows that the transshipment policy has the lowest share in all scenarios. This result could be a subject for future research since transshipment or mode shift plays an important role in synchromodal transport. The other policy in this model is to wait, which is essentially the traditional reaction, and detour, which is practically difficult for barges and/or freight trains.

More recent studies integrate a learning approach within the synchromodal framework the potential of this approach to handle the uncertainty in the nature of disruptions in the network. A study by Guo et al. (2022) that adopts the RL approach in an offline setting to

solve the global shipment matching problem under dynamic and stochastic travel time settings. Another study using the RL technique under the synchromodal framework is proposed by Zhang et al. (2023). This study combines an Adaptive Large Neighborhood Search (ALNS) in Zhang et al. (2022) with RL to address the service time uncertainty in different operations in synchromodal transport. It incorporates an event trigger mechanism in case of an unexpected event with an uncertain duration. Compared to these studies, our approach proposes a cost-based reward mechanism and accounts for shipment volume and the length of the delay in determining the reward. Additionally, we have integrated a discrete-event simulation with RL as a training environment for the RL agent to learn from experience during training. Consequently, exposing the RL agent to an environment that closely mirrors real-world operations, including disrupted scenarios, is expected to yield distinct RL outcomes. As the closest modeling approach to this study is that of Zhang et al. (2023), it has been used as a benchmark for evaluating our results.

Based on the identified research gaps in relying on deterministic or stochastic optimization methods that assume full information availability a priori, simplified or static disruption modeling, lack of interactive training environments, and insufficient focus on cost trade-off and practical constraints such as delay length, and considering limited use of RL, this research aims to:

- Develop a hybrid simulation-optimization framework that integrates RL for effectively managing disruptions in synchromodal freight transportation systems.
- Model the stochastic nature of disruptions in freight transport networks through discrete-event simulation to create realistic representations of both supply-side and demand-side disruptions.
- Design and evaluate decision support mechanisms using RL that balance trade-offs among cost components, including transport, storage, delay penalties, and handling costs, under various disruption scenarios. This is intended to resolve unforeseen disrupted situations and enable resolving those in time and efficiently.
- Compare and analyze the performance of RL-based disruption management policies against benchmark approaches to demonstrate their resilience and cost-effectiveness in synchromodal networks.
- Test the proposed framework on a real-world case study to validate its practicality, scalability, and effectiveness in managing disruptions and improving overall system performance.

This paper advances the proposed model by Zhang et al. (2023) in several ways, including a comprehensive disruption profile, integrating a discrete-event simulation, and proposing a negative cost value as the reward. The latter addition considers higher additional costs due to disruptions for higher shipment volumes and more severe disruptions, allowing the RL agent to prioritize larger shipments. Extensive disruption scenarios allow for an improved learning process. The disruptions impact demand and services, a feature that has not been extensively studied in the literature. The output, in the form of a modular framework, offers a plug-and-play mechanism that allows for improving/replacing/extending the modules as needed as a valuable asset for researchers and industry alike.

## 3. Problem description

The focus is on port-hinterland freight transportation, specifically the unidirectional flow of shipments from the main port to various inland terminals, excluding the final leg to distribution centers or warehouses as illustrated in Fig. 1. Each terminal is connected by service lines, where each line represents a service via barge, train, or truck fleet. Shipments can be transported directly or across multiple service lines, involving transfers at transshipment terminals, thus forming a multimodal transport network. Real-time information sharing among stakeholders is assumed, enabling the central planner to flexibly reroute shipments, provided the necessary Information and Communication Technology (ICT) infrastructure is in place. Additionally, modal-free booking applies to all shipments, granting the planner full authority to reallocate containers as needed.

In the planning phase, shipments are assigned to itineraries comprising one or multiple service lines selected according to specific shipment requirements. When disruptions occur, the re-planning is triggered to propose alternative itineraries to mitigate the disruption. The decision needs to be made between two options: (1) staying in the original itinerary designated during the planning phase, referred to as ‘wait’, or (2) reassigning the shipment to the alternative itinerary, referred to as ‘reassign’. These mitigation strategy is presented in Fig. 2. In real-world, this decision is often made based on an estimated duration of disruptions according to the best knowledge of the operator as this duration is subject to variations. To deal with this uncertainty, In several cases, the shipment may continue following the original itinerary despite disruptions as the additional waiting costs may be lower than the costs associated with selecting an alternative route.

The wait or reassign option is evaluated by a learning agent using RL. The agent selects the optimal action upon the occurrence of disruptions, adapting to the current state of the environment to minimize additional costs incurred. This approach allows the agent to learn and improve decision-making over time, enhancing resilience against disruptions.

## 4. Methodology

To address the objective of the paper, i.e. to provide a decision-support tool for managing disruptions in the synchromodal framework, the proposed methodology comprises the identification of disruption profiles and the development of a learning-based simulation-optimization framework that integrates the disruption profiles in the modeling process. In the context of this research, disruptions refer to events that cause deviations from planned schedules, affecting either ’services’ or ’requests’. These disruptions can result in service delays, reduced capacities, or adjustments in shipment release time. The random nature of these disruptions follows

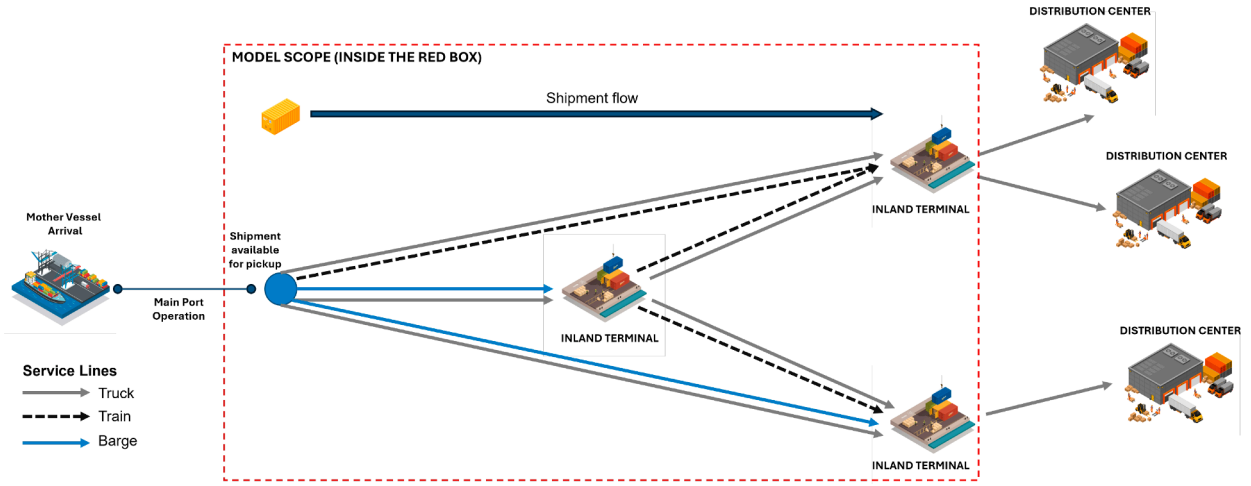



**Fig. 1.** Model scope illustration.

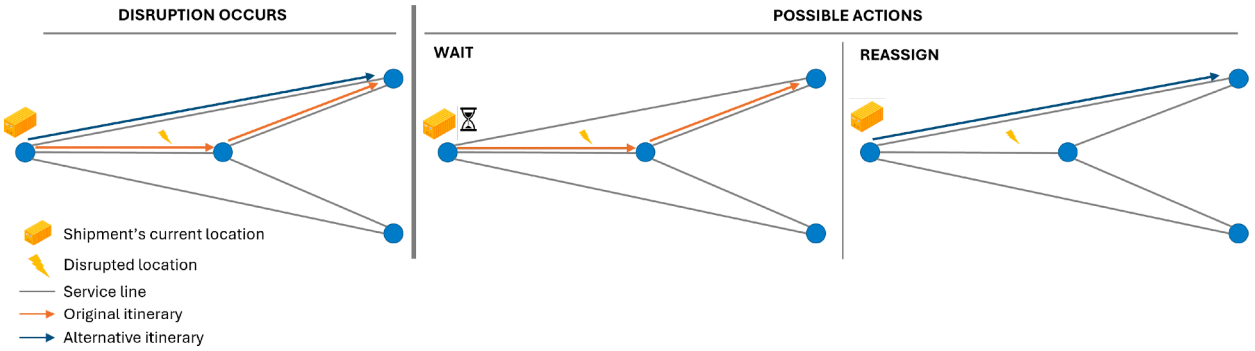



**Fig. 2.** Disruption mitigation strategies.

the profiles outlined in Tables 1 and 2, impacting specific service lines, terminals, or requests. The modeling framework consists of three main modules: a Simulation Module, which serves as the dynamic environment; an Optimization Module acting as the central planner to provide shipment matching plans; and a Learning Agent, which represents the disruption manager providing decisions during the disruption occurrences. The modeling framework therefore enables us to realize the adaptive nature of synchromodal transport systems with the developed modules. To be precise, the adaptability nature is incorporated at the demand level, allowing containers to be reallocated across different services. This flexibility is essential for real-world applicability, as alternative approaches may be challenging to implement in practice. Moreover, the use of RL is justified and necessary to handle the high level of uncertainty in disruptions, where simple assumptions about their impact and frequency as predefined distributions of the duration of the disruption are insufficient. In real-world operations, disruption duration is often estimated based on the best knowledge of the operator, yet it remains highly variable.

The remainder of this section first presents the identification of the disruption categories and then the three building blocks of the modeling framework.

### *4.1. Disruption categorization*

Regardless of the source, disruptions in the freight network vary in type and impact and may require different reaction strategies at strategic, tactical, or operational levels. Such events can be distinguished according to frequency and severity, categorized into endogenous and exogenous factors (Hrušovský et al., 2021). These disruptions may arise from various sources, including natural causes or human actions (Wide et al., 2022). For instance, some disruptions may have a low probability of occurrence but cause severe impacts, while others may occur frequently but with lower consequences. Categorizing disruptions in this way helps simplify modeling while maintaining realistic system behavior. Additionally, it serves as a foundation for developing strategies by determining which policies are best suited to specific disruption categories.

In this research, disruptions in the freight network are divided into two main categories comprising those affecting the service network (supply side) and those impacting shipment requests (demand side). Each disruption profile represents a distinct group of related disruptions, allowing for a more structured analysis of their effects.

**Table 1**
Service disruption profile.

| Profile | Description | Mode/ Location | Effect in the Simulation | Duration | Capacity Reduction | Occurrence per Year |
|---|---|---|---|---|---|---|
| 1 | Operational delays, road congestions | Train, Truck | Delay | 1–3h | 0 % | 30 % |
| 2 | Operational delay, canal congestion | Barge | Delay | 1–6h | 0 % | 35 % |
| 3 | Bad weather, labor strike, accident, systems maintenance | Train, Barge | Delay | 12–48h | 0 % | 6 % |
| 4 | Terminal congestion, operational delay | Terminal | Delay | 1-3h | 0 % | 30 % |
| 5 | High and low water level | Barge | Carrying capacity reduction | 12–24h | 15–20 % | 10 % |

**Table 2**
Request disruption profile.

| Profile | Description | Location | Effect in the simulation | Delay Release | Valume Change | Occurrence per Year |
|---|---|---|---|---|---|---|
| 6 | Demand Change | Shipment | Volume change | – | –30 % to + 30 % | 30 % |
| 7 | Customs issues, main port operational delays | Shipment | Release time change | 1–6h | – | 30 % |
| 8 | Mother vessels arrival delays | Shipment | Release time change | 1–7d | – | 5 % |

On the supply side, five distinct profiles are determined. These include frequent disruptions that may lead to operational delays, which can be caused by road congestion (Ambra et al., 2019) or train-related issues (Palmqvist et al., 2022). Additionally, delays in barge service can occur due to congestion in rivers caused by locks or high traffic (Barkley and Mcleod, 2022). More severe disruptions, such as bad weather or system maintenance, can have a prolonged impact, resulting in operations being halted for an extended period (Ambra et al., 2019). Another profile considers disruptions in nodes, such as terminals, including equipment problems or congestion (PoR, 2024). Lastly, reductions in barge carrying capacity can occur due to fluctuations in river water levels (van Dorsser et al., 2020; CCNR, 2020), which restrict barges from carrying containers at full capacity. Low water levels may force barges to reduce their draft, while high water levels can limit the height of stacked containers to prevent collisions with bridges.

On the demand (request) side, three disruption profiles are considered, including two profiles related to deviations in container release times and one profile concerning alterations in shipment volume. In port-hinterland transportation, changes in the release time could occur due to various reasons such as late arrival of the mother vessel which can delay the release time up to seven days (Statista, 2022) or relatively minor issues such as customs clearance where the delay could be less than a day. Meanwhile, changes in the transport volume may arise from experiencing unexpected increases in demand that exceed long-term contracts. The request disruption profiles are presented in Table 2.

The disruption profiles are created based on two spectra described by Wide et al. (2022). Disruptions characterized by high probability and low severity, indicated by high yearly occurrence rate and low severity (columns 5 to 7) are represented in Profiles 1, 2, 4, 6, and 7. Disruptions with low probability and high severity are attributed to Profiles 3, 5, and 8. It should be noted that each profile can occur only in certain locations such as a terminal, a service mode, or directly on the shipment, as indicated in column 3.

### *4.2. Simulation module*

The simulator's primary role is to model the dynamic nature of disruptions in the hinterland freight network, closely mirroring real-world operations. Without loss of generality, inland terminals are represented by nodes, with one or more nodes within the main port acting as origin points where shipments are loaded onto transport modes. Additional nodes are distributed throughout the hinterland, functioning as transshipment terminals or destination points. Each terminal is characterized by a handling capacity that impacts loading and unloading times. In this simulation, parameters such as stacking yard capacity and vehicle buffer area are assumed to be infinite. Exceeding these capacities will typically lead to terminal congestion and potential delays. However, the scope of this paper does not include simulating detailed port operations. Instead, port congestion is represented by random variables impacting the change in release time of a shipment to model unexpected events.

The simulation module has three main components for simulating the behavior of *services*, *shipments*, and *disruptions*. Services operate on either fixed or flexible schedules; fixed schedules are predefined, while flexible schedules adapt based on assigned requests. Service processes are defined by key operational parameters for each mode, including travel speed, carrying capacity, origin and destination, and departure time. These parameters are crucial in determining event occurrences within the discrete event simulation.

Each shipment comprises a bundle of containers under a single contract, sharing the same origin and destination. Consequently, each request is characterized by an origin, a destination, and a container volume. Time parameters, including announcement time, release time, and due time, are also specified for each request. The due time reflects the customer's expectations, with any delay beyond this time incurring a penalty. The shipment and services follow parallel processes while continuously interacting with each

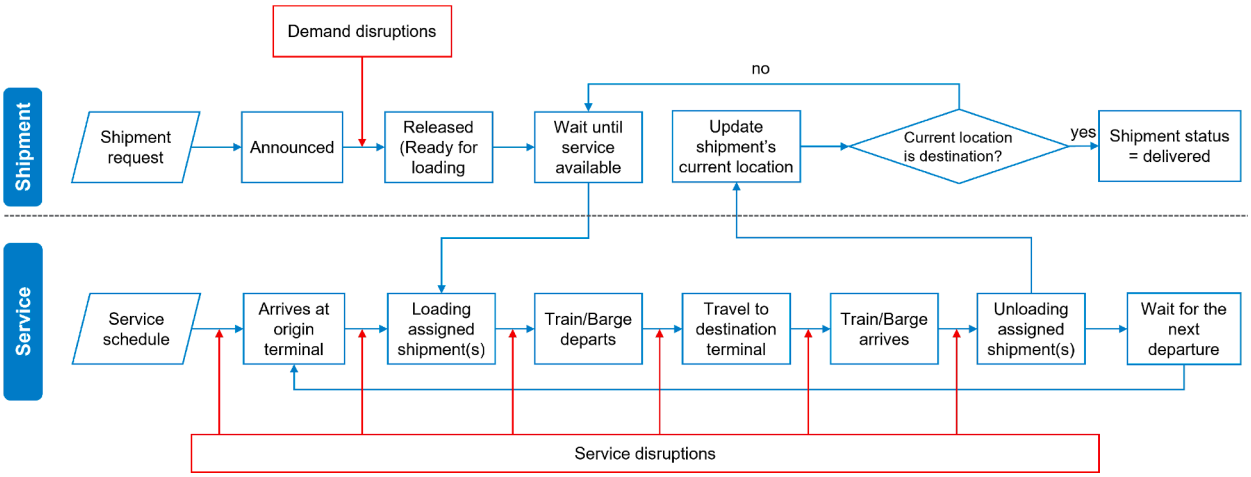



**Fig. 3.** Simulation flow process.

other. The simulation records the actual costs in a cost unit incurred for transporting each shipment from origin to destination, with cost components covering storage, handling, travel, and delay penalties.

During the simulation process, disruptions are introduced based on predefined profiles. Disruptions affecting shipment requests are limited to a narrow time window-specifically, after a shipment is announced but before it is released, causing a change in the release time or volume of the affected shipments. In contrast, disruptions impacting service lines and terminals can occur at any point within the event chain, causing the affected services to pause until the disruption is resolved. Following a disruption and in the absence of synchromodal settings, the request assigned to a disrupted service will wait until the service resumes operations. This is referred to as the *always wait* policy.

The simulation flow in Fig. 3 represents the interaction between shipments, services, and disruptions.

### 4.3. Optimization module: hybrid simulation-optimization

The role of the Hybrid Simulation-Optimization (HSO) module is to match shipment requests with available services while accounting for capacity limits, as well as time and cost parameters associated with each service line, especially in the event of disruptions. The planning is carried out using an optimization model on a periodic (e.g. weekly basis) and remaining shipments are served in the subsequent planning horizon. The planning module is integrated with the simulator and is triggered additionally whenever a disruption occurs, i.e. on an event-based rolling horizon basis. Such mechanism is enabled by the development of Information and Communication Technologies (ICT), allowing for information exchange and system planning in real-time Di Febbraro et al. (2016). Then, affected requests are identified based on the location and time of the disruption. It is important to distinguish between the 'affected' and 'disrupted' requests. Affected requests are those that can still be re-routed despite disruptions. However, a shipment is excluded from the set of affected requests in three specific scenarios where rerouting is ineffective. First, if the shipment is located at a terminal or onboard a service line where the disruption has occurred. Second, if the shipment is on a service line and a disruption affects its destination terminal. Third, if the disruption impacts the shipment's final destination. In these cases, replanning cannot enable the shipment to avoid the disruption.

The HSO module introduces an 'always reassign' policy to complement the 'always wait' policy discussed in the previous section. When a new request is received or an affected shipment is detected, the optimization module is triggered to initiate replanning. This process involves assigning an optimal service line to each shipment. In the case of replanning, the updated assignment overrides the original itinerary, enabling affected shipments to be redirected to alternative service lines. The disrupted location is excluded from the potential solution space, ensuring efficient reassignment. To achieve this, the deterministic optimization model, as adapted from Filom and Razavi (2025) and Guo et al. (2021), is integrated within the HSO module. The HSO module receives shipment requests and available service lines as input, using a path-based approach combined with a preprocessing algorithm to streamline path generation and reduce computational load. This algorithm identifies feasible paths by assembling service combinations that meet the spatiotemporal requirements and ensure transshipment feasibility.

The Objective Function 1 considers each service line's departure and arrival times. It maximizes the number of matched requests (first term) and minimizes the total cost including transport (second term), transshipment (third term), storage (fourth term), delay (fifth term), and handling (last term) costs. Given the known departure and arrival times, storage time is calculated based on the service schedule of each transport mode in the path, ensuring accurate cost and time estimations for each feasible route. The notations are presented in Table 3. The objective function is subject to more than 15 constraints which can be grouped into several categories. Request acceptance constraints ensure that requests are only accepted if there are available services at both the origin and destination. Matching constraints guarantee that each request is paired with at most one service for both departure and arrival. Flow

**Table 3**
Notation.

| Sets and indices | |
|---|---|
| $R$ | Request ($r \in R$) |
| $R^t$ | Requests during time interval $(t-1, t]$ ($r \in R^t$) |
| $S$ | Services, $S = S_{\text{marine}} \cup S_{\text{rail}} \cup S_{\text{truck}}$ |
| $S_m$ | Services with mode $m \in M$ |
| $S_i^+$ | Services departing at intermodal terminal $i \in I$, $I_i^+ = S_{i,\text{marine}}^+ \cup S_{i,\text{rail}}^+ \cup S_{i,\text{truck}}^+$ |
| $S_i^-$ | Services arriving at intermodal terminal $i \in I$, $S_i^- = S_{i,\text{marine}}^- \cup S_{i,\text{rail}}^- \cup S_{i,\text{truck}}^-$ |
| $S_i^{+t}$ | Services departing at origin intermodal terminal $i \in I$ during time interval $(t-1, t]$ |
| $S_i^{-t}$ | Services arriving at destination intermodal terminal $i \in I$ during time interval $(t-1, t]$ |
| $I$ | Terminals ($i \in I$) |
| **Parameters** | |
| $v_r$ | Penalty for unmatched request |
| $c_s$ | Travel cost for path $s$ |
| $D_s$ | Scheduled departure time of service $s \in S$ |
| $f_i^L$ | Loading/unloading time per container at terminal $i \in I$ |
| $U_s^t$ | Free capacity of service $s \in S$ at decision epoch $t$ |
| $u_r$ | Volume for request $r$ |
| $c_{ri}^T$ | Transshipment cost for requests $r$ in Terminal $i$ |
| $c_i^S$ | Storage cost in Terminal $i$ |
| $\tilde{S}_{ri}^s$ | Storage time of request $r$ in Terminal $i$ for path $s$ |
| $c_r^D$ | Delay penalty for request $r$ |
| $c_{ri}^L$ | Handling cost for request $r$ in node $i$ |
| **Variables** | |
| $x_{rs}^t$ | Binary variable for matching between request $r$ and path $s$ in time $t$ |
| $y^t$ | Binary variable for unmatched request in time $t$ |
| $\tilde{d}_{ri}^r$ | delay time of request $r$ in Terminal $i$ |
| $z_{rsp}^t$ | Binary variable; 1 if request $r \in R$ is matched with service $s \in S$ and transshipment service $p \in S$ at decision epoch $t$, $x_{rs} = 1$, and $x_{rp} = 1$, 0 otherwise |

and transshipment constraints prevent subtours, maintain flow conservation at terminals, and ensure transshipments occur logically and within time limits. Capacity and time feasibility constraints check that service capacities are not exceeded and that loading, departure, and transshipment times align with request schedules. Cost-related constraints calculate loading, unloading, transshipment, and storage costs across all terminals. Finally, delay constraints account for penalties due to late arrivals at the destination terminal. Due to readability considerations and to avoid repetition, not all equations are presented here.

Constraints 2 and 3 ensure that a request is accepted by the platform only if there are available services departing from the request's origin and arriving at its destination, respectively. Constraints 4 and 5 enforce that each request $r \in R^t$ is matched with at most one service for a given origin and destination. Constraints 6 to 9 eliminate subtours from the solution. Specifically, constraints 7 and 8 prevent subtours from forming at the shipment's origin and destination, while constraints 8 and 9 ensure that each itinerary includes exactly one origin and one destination. Constraint 10 maintains flow conservation at transshipment terminals. Constraint 11 ensures that the total number of containers assigned to a service $s \in S$ does not exceed its available capacity during decision epoch $t \in T$. Constraint 12 checks the temporal feasibility of matching a request to a service by requiring that the service's departure time, adjusted for loading time based on container volume, is no earlier than the request's release time. The parameter $B$ is a large constant used to validate the constraint when a request is matched to a service (i.e., $x_{rs}^t = 1$). Constraints 13 to 15 govern the transshipment logic using binary variables $x_{rp}^t$ and $z_{rsp}^t$. The variable $x_{rp}^t$ indicates whether a potential transshipment service $p \in S$ is matched at $d_s$, with $d_p$ matching the request's destination $d_r$. The variable $z_{rsp}^t$ equals 1 only when both $x_{rs}^t = 1$ and $x_{rp}^t = 1$, indicating that a transshipment has occurred between services $s$ and $p$. Constraint 16 enforces temporal feasibility for transshipments at intermediate terminals. Constraints 17 and 18 compute loading and unloading costs at the request's origin and destination, respectively. Constraint 19 calculates the combined loading and unloading costs at the transshipment terminal. Constraints 20 to 22 handle the calculation of storage costs: 20 for the origin terminal, 21 for the transshipment terminal, and 22 for the destination terminal. Finally, constraint 23 determines the delay time at the request's destination intermodal terminal.

$$\min_{x^t, y^t} \left( \sum_{r \in R} v_r (1 - y^t) + \sum_{r \in R} \sum_{s \in S} c_s x_{rs}^t u_r + \sum_{r \in R} \sum_{s \in S} c_{ri}^T x_{rs}^t u_r + \sum_{r \in R} \sum_{i \in I} c_i^S x_{rs}^t \tilde{S}_{ri}^s u_r + \sum_{r \in R} c_r^D x_{rs}^t \tilde{d}_{ri}^r + \sum_{r \in R} c_{ri}^L x_{rs}^t u_r \right) \tag{1}$$

Subject to:

$$y_r^t \leq \sum_{s \in S_{o_r}^+} x_{rs}^t, \quad \forall r \in R^t, \tag{2}$$

$$y_r^t \leq \sum_{s \in S_{d_r}^-} x_{rs}^t, \quad \forall r \in R^t \tag{3}$$

$$\sum_{s \in S^+_{o_r}} x^t_{rs} \leq 1, \quad \forall r \in R^t \tag{4}$$

$$\sum_{s \in S^-_{d_r}} x^t_{rs} \leq 1, \quad \forall r \in R^t \tag{5}$$

$$\sum_{s \in S^-_{o_r}} x^t_{rs} \leq 0, \quad \forall r \in R^t \tag{6}$$

$$\sum_{s \in S^+_{d_r}} x^t_{rs} \leq 0, \quad \forall r \in R^t \tag{7}$$

$$\sum_{s \in S^+_i} x^t_{r,s} \leq 1, \quad \forall r \in R^t, i \in I \setminus \{o_r, d_r\} \tag{8}$$

$$\sum_{s \in S^-_i} x^t_{r,s} \leq 1, \quad \forall r \in R^t, i \in I \setminus \{o_r, d_r\} \tag{9}$$

$$\sum_{s \in S^+_i} x^t_{r,s} = \sum_{s \in S^-_i} x^t_{r,s}, \quad \forall r \in R^t, i \in I \setminus \{o_r, d_r\} \tag{10}$$

$$\sum_{r \in R^t} x^t_{r,s} u_r \leq U^t_s, \quad \forall s \in S \tag{11}$$

$$D_s + B(1 - x^t_{rs}) \geq a_r + f^L_i u_r, \quad \forall r \in R^t, s \in S \tag{12}$$

$$z^t_{rsp} \leq x^t_{rs} \quad \forall r \in R^t, s \in S, p \in S \tag{13}$$

$$z^t_{rsp} \leq x^t_{rp} \quad \forall r \in R^t, s \in S, p \in S \tag{14}$$

$$z^t_{rsp} \geq x^t_{rp} + x^t_{rs} - 1 \quad \forall r \in R^t, s \in S, p \in S \tag{15}$$

$$D_s + t_s + 2f^L_i \leq D_p + B(1 - z^t_{rsp}), \quad \forall r \in R^t, s \in S, p \in S, i \in I \setminus \{o_r, d_r\} \tag{16}$$

$$c^L_{ri} = \sum_{s \in S^+_i} c^L_i x^t_{r,s}, \quad \forall r \in R^t, i = o_r \tag{17}$$

$$c^L_{ri} = \sum_{s \in S^-_i} c^L_i x^t_{r,s}, \quad \forall r \in R^t, i = d_r \tag{18}$$

$$c^T_{ri} = \sum_{s \in S^+_i} \sum_{p \in S^-_i} c^T_i z^t_{rsp}, \quad \forall r \in R^t, i \in I \setminus \{o_r, d_r\} \tag{19}$$

$$\tilde{S}^s_{ri} = max(0, x^t_{rs}(D_s - a_r - f^L_i u_r)) \quad \forall r \in R^t, s \in S, i = o_r \tag{20}$$

$$\tilde{S}^p_{ri} = max(0, z^t_{rsp}(D_{rp} - D_{rs} - t_s - 2f^L_i)) \quad \forall r \in R^t, s \in S, p \in S, i \in I \setminus \{o_r, d_r\} \tag{21}$$

$$\tilde{S}^s_{ri} = max(0, x^t_{rs}(e_r - D_s - t_s - f^L_i)) \quad \forall r \in R^t, s \in S, i = d_r \tag{22}$$

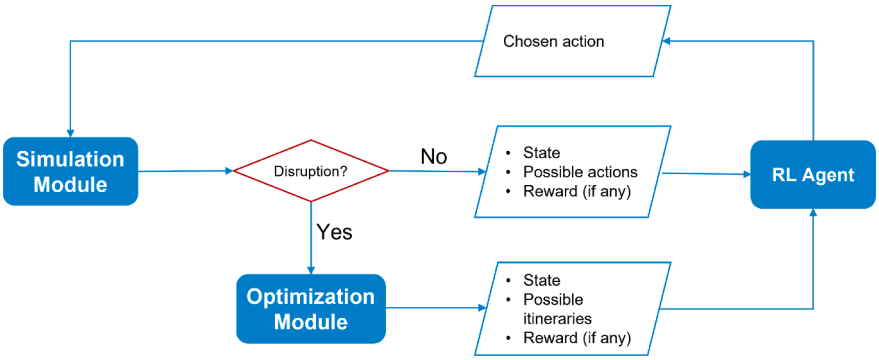



**Fig. 4.** Markov decision process in learning assisted hybrid simulation-optimization.

$$\tilde{d}_{ri}^{s} = max(0, x_{rs}^{t}(A_s + f_i^L - e_r)) \quad \forall r \in R^t, s \in S, i = d_r \tag{23}$$

The model employs a K-best solution approach to improve computational efficiency required for training the model. As a result, $k$ alternative solutions are generated to serve as backup itineraries during the planning process and sort the solutions from the best to the worst according to the objective function. The simulation algorithm discards disrupted itineraries and selects the best option from the remaining viable solutions. Each time a shipment departs from a terminal, any solutions that diverge from that terminal are also eliminated. If no viable backup itineraries remain for a shipment, the optimization algorithm is re-triggered to generate new options. This method minimizes the frequency of optimization triggers during replanning, reducing overall runtime. However, it may yield suboptimal solutions, as it does not fully consider real-time constraints. To analyse this, this approach has been compared against the alternative (always considering the optimization model) in Section 5.

### *4.4. Learning-based decision making module*

Reassigning a shipment to an alternate service line during disruptions can enhance the resilience of the freight network, aligning well with the flexibility of a synchromodal framework. The decision to either 'wait' or 'reassign' a shipment in the event of a disruption is highly dependent on disruption characteristics. To address the uncertainty involved, a learning agent (RL agent) trained via RL is integrated to determine the optimal decision between waiting and reassigning. This approach enables the model to make more informed, adaptive decisions, balancing the advantages of the reassignment policy against the potential benefits of the waiting policy.

Using a value function, the RL agent selects the best action for a given state by learning from past experiences and extensive training. Fig. 4 shows the Markov Decision Process (MDP) providing the framework for the RL agent to interact with its environment. After disruptions occur, the optimization module is triggered and provides options for the RL agent to choose from. However, disruptions on requests do not engage the RL agent, as shipment properties (e.g., release time or volume) are adjusted immediately, assuming the disruption's duration is known in advance. During the service disruption, the duration is unknown, requiring the RL agent to assist the model in selecting actions based on the given states. The action space for the RL agent in this model consists of two actions: reassign and wait.

From the perspective of a centralized planner in synchromodal transport, defining the state of the RL agent is complex. The state space becomes exceedingly large if all shipments and service attributes are considered. However, by narrowing the perspective to a single shipment, defining the state becomes more manageable, given that each request has an independent decision-making process. In this approach, the state consists of six features: the request's current position, destination, due time, volume, type of disruption, and the current time.

The optimal policy is determined for each shipment independently. The action space is modeled by treating the itinerary of each shipment as a series of actions, where each action represents a single service line. For instance, assuming a shipment is assigned to service lines [Barge1, Train1], possible actions during a disruption could include: *wait* ([Barge1, Train1]) or *reassign* ([Truck2]). If the RL agent selects *wait*, the shipment will proceed with Barge1. After arriving at Barge1's destination, one action is completed and the simulation module updates the state of the shipment, grants the reward, and takes the next action Train1. At this stage, no further decision-making occurs unless a disruption arises, as only one action (continuing with Train1) is available. Decisions for each shipment and the corresponding actions are stored, enabling subsequent actions to be informed by previous ones and ensuring interdependency of decision-making.

For each shipment, the RL agent uses a reward, comprising transport, storage, handling, and delay actual costs, to update the action value function $Q(s, a)$ for each state $s$ and action $a$ using the off-policy Temporal Difference Control, specifically the Q-learning technique (Eq. 24). The algorithm considers the reward $R_{t+1}$ of the next action, the maximum next action values $Q(S_{t+1}, a)$ that are discounted by $\gamma$, and the learning rate $\alpha$. Unlike *Monte Carlo*, Q-Learning does not have to complete one full episode to update the action value function. Instead, it uses the current estimate in the equation. This method proves to reach convergence faster (Sutton and Barto, 2018). When a disruption impacts a shipment and continues until either action is completed (the shipment arrives at the

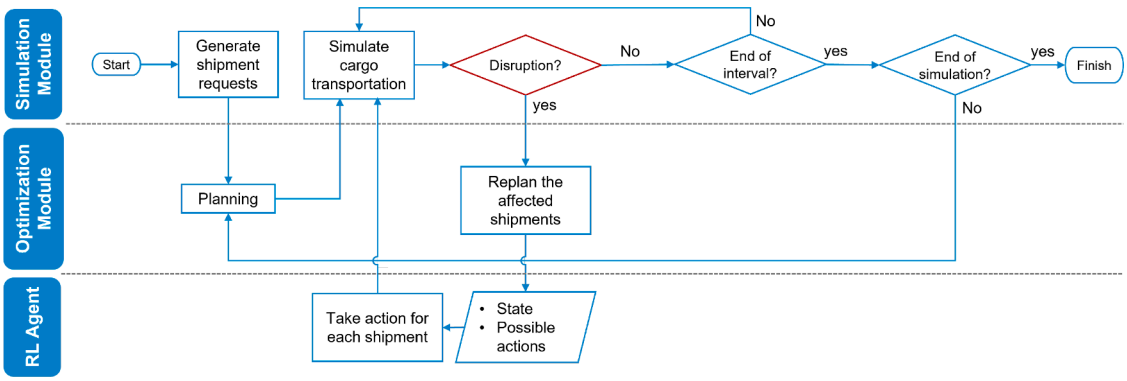



**Fig. 5.** Learning-assisted hybrid simulation-optimization flow.

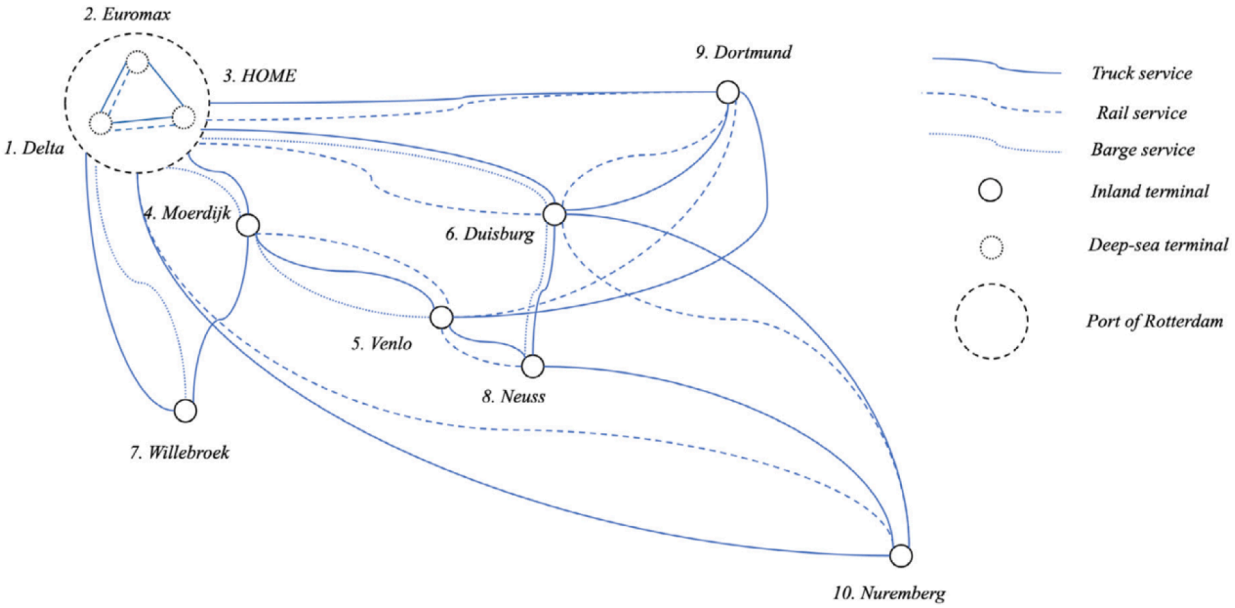



**Fig. 6.** EGS service network. source: (Zhang et al., 2023).

next destination) or another disruption affects the same shipment. Since rewards are negative costs, the agent selects actions based on the value function, choosing those with values closest to zero to minimize costs.

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\left[R_{t+1} + \gamma \max_a Q(S_{t+1}, a) \quad - Q(S_t, A_t)\right] \tag{24}$$

The learning technique is integrated with hybrid simulation-optimization to create a comprehensive decision support system for a resilient synchromodal framework. These three modules are interconnected by exchanging information, as illustrated in the flow diagram in Fig. 5, forming a complete model of Learning-Assisted Hybrid Simulation-Optimization within the synchromodal framework.

## 5. Case study

For the case study, the service network is adapted from the European Gateway Services (EGS) network, now known as Hutchinson Ports Europe Intermodal (HPEI) and requests are synthetically generated. This network connects the Hutchinson Ports ECT in the Port of Rotterdam to inland terminals in the Netherlands, Belgium, and Germany through the Rhine-Alpine corridor. The details of the network, including distances between terminals and the service schedule, are based on the study by Zhang et al. (2023).

The network shown in Fig. 6 consists of 10 terminals, with three located in the Port of Rotterdam and the remaining seven spread across the hinterland as inland terminals. The service network includes 49 barges, 33 trains, and 34 truck service lines operating between these terminals. Service lines representing scheduled services connect an origin-destination (OD) pair with a unique weekly departure time. Service frequency for the same OD pair and mode is indicated by multiple lines. Each service line represents a

transport mode that connects two nodes. Flexible services (such as trucks that do not have a schedule) are represented by one line for each origin-destination pair and are assumed to have an unlimited capacity.

Additionally, a carrying capacity of 160 TEU and 90 TEU is assumed for barge and train services, respectively; the travel speed is set to 15 km/h, 45 km/h for trains, and 75 km/h for barges; and the distances between terminals vary depending on the service mode.

Requests are generated randomly according to the service capacity proportion of each OD pair. For instance, given the higher number of services connecting Delta to Nuremberg compared to those connecting Delta to Willebroek, the request volume from Delta to Nuremberg is proportionally higher. Requests are announced in batches, with each batch comprising multiple shipment requests. For the default scenario, 200 requests are generated and announced proportionally over 3 weeks. This default demand scenario is used to train the RL agent.

### 5.1. Model training

For the purpose of training, the default demand scenario comprising 200 requests is used and requests are distributed over 3 weeks, assuming a weekly demand of 60 to 70 requests.

As explained in Chapter 4, the HSO model uses a path-based approach. Each path consists of one or possible combinations of multiple service lines connecting certain origins to destinations. To generate paths, the travel cost for each service line is determined using two key travel cost parameters: time- and distance-related costs, considering the properties of each service line.

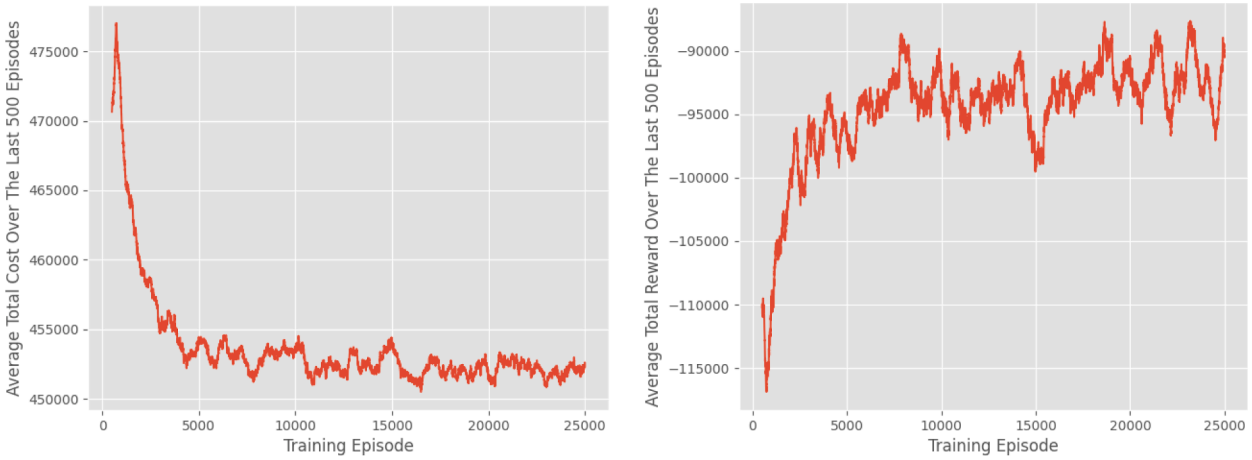
The simulation is set to 5-weeks to ensure completion of all delivery requests, including those experiencing significant delays. This extended time frame enables more representative and comprehensive results for all delivery processes. Each training episode uses a unique random seed to introduce variability in disruptions, ensuring diverse scenarios.

During training, only service disruptions are considered, as demand disruptions do not activate the RL agent. This approach ensures the agent focuses on scenarios that directly impact its decision-making process. Lastly, the model undergoes 25,000 episodes of training, and the action-value function, represented as a Q-table, is extracted at intervals of 5000 episodes for use in results analysis. Each episode introduces randomized disruptions, exposing the agent to diverse scenarios and reducing the risk of overfitting. Three policies of *always wait* (AW), i.e. no re-planning, *always reassign* (AR), i.e. re-planning by triggering the optimization, and *RL-based*, i.e. involving the RL agent to find the *optimal policy* are evaluated.

To balance the exploration and exploitation mechanisms (Sutton and Barto, 2018), in the training phase, the $\epsilon$ value provides a small probability of selecting an action with a lower value in order to explore in order to explore and achieve a higher reward in the longer run. In the implementation phase, the RL agent always selects actions with a higher value in the given states. For training, $\epsilon$ is set to 0.05, while the discount factor ($\gamma$) and learning rate ($\alpha$) are set to 0.9 and 0.5, respectively.

### 5.2. Numerical experiments against benchmarks

The performance of the RL-assisted model is compared against benchmark policies (AW and AR). To this aim, each policy is simulated across multiple episodes with randomized disruptions, showcasing the response, resilience, and adaptability of policies in response to diverse and unpredictable conditions. In this experiment and to test the performance, the RL-assisted model is simulated five times (i.e. by varying the Q-table extracted from 5000 to 25,000 training episodes) using the same simulation settings of 100 testing episodes and disruption profiles.




**(a)** Total Cost - Episodic Training **(b)** Total Reward - Episodic Training

**Fig. 7.** Episodic training for 25,000 episodes.

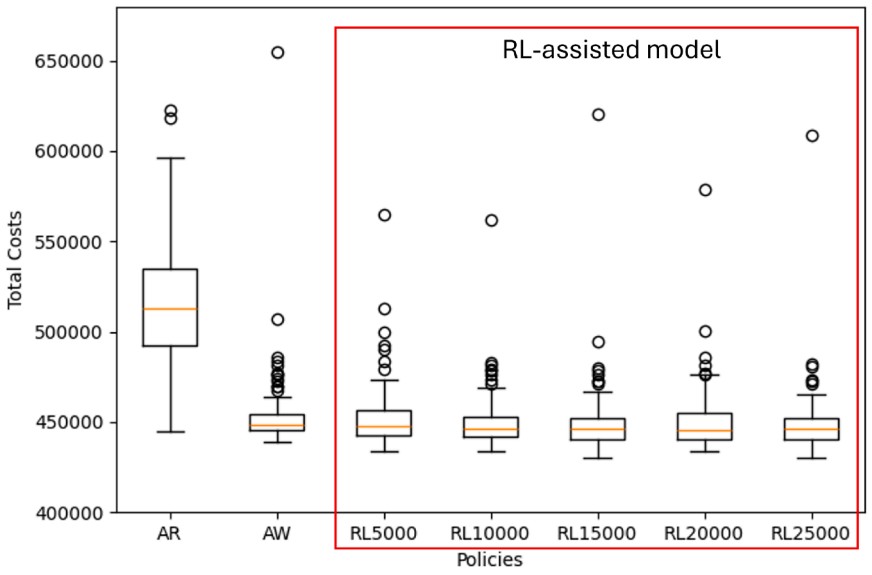



**Fig. 8.** Policy comparison (multiple cases) - AW:Always wait, AR: Always reassign, RL{NUMBER}: the number of training episodes.

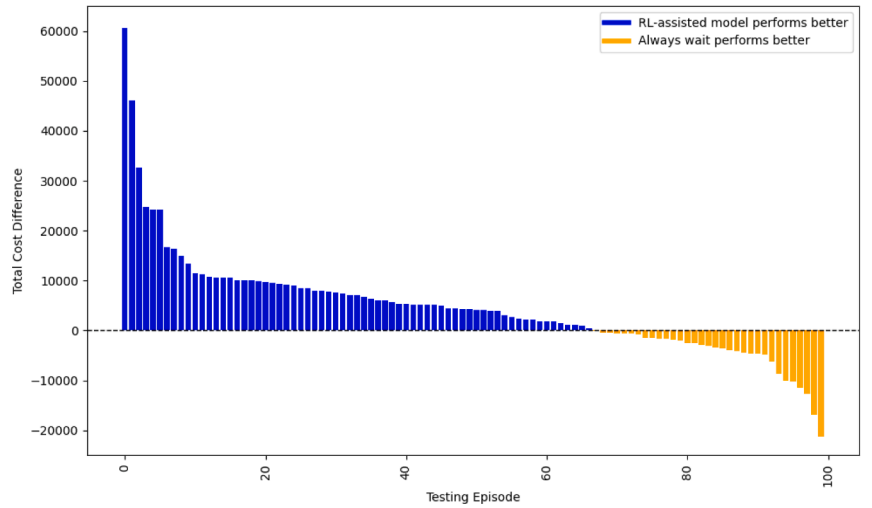
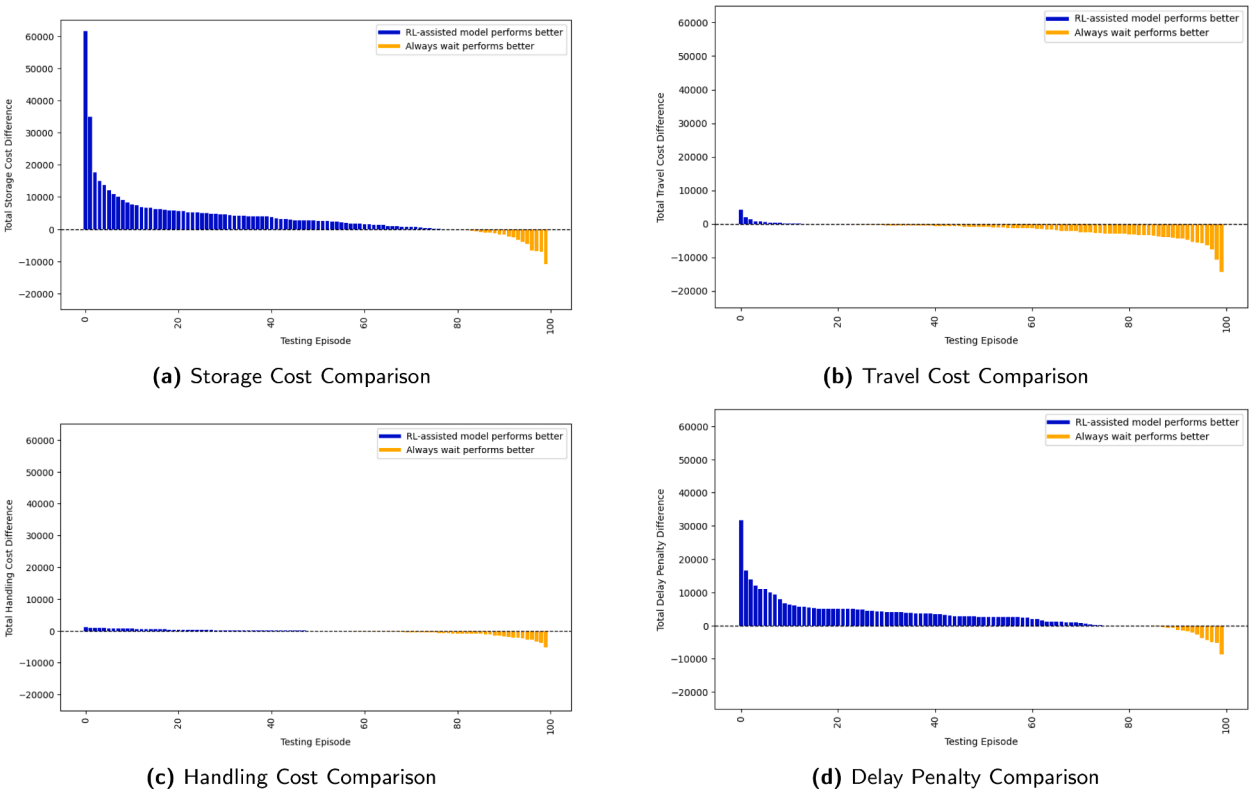



**Fig. 9.** Comparison between always wait and RL-assisted model in episodic simulations.

Fig. 7 presents the result of the RL agent training over 25,000 episodes with the rolling mean applied using a 500-episode window. At about 10,000 training episodes, the improvement in mean total costs and rewards declined compared to the early training stages. Nonetheless, the training continues to expose the RL agent to an extended range of experiences, resulting in a resilient model able to react to various disruption scenarios.

Fig. 8 shows the total costs of the system under each policy. RL{NUMBER} indicates the number of training episodes when the Q-table is extracted. The AW policy significantly outperforms the *always reassign* policy with the given disruption profile set. However, it shows some outliers with high costs. Since the majority of disruption profiles exhibit low severity, the wait action generally yields better outcomes than reassignment. In one complete simulation, AR might generate a high total cost due to several 'reassign' action on certain shipments that have a significantly higher cost compared to 'wait' action. However, in certain cases, reassignment may still be beneficial, as demonstrated in the next analysis (Fig. 9), where the RL agent outperforms AW by combining wait and reassign actions based on the situation. This is further clarified in Fig. 15, which compares model performance under severe and short delay disruptions. In the severe disruption setting, AR performs much closer to AW, as reassignment becomes more advantageous. Notably, the RL agent learns to select reassign actions in specific states, leading to improved performance over both benchmark policies. Additionally, the K-best approach provides solutions in advance, which can result in suboptimal alternative solutions for reassignment action. The optimality gap between K-best solutions and optimal solutions will be discussed later in this section.




**(a)** Storage Cost Comparison

**(b)** Travel Cost Comparison

**(c)** Handling Cost Comparison

**(d)** Delay Penalty Comparison

**Fig. 10.** Cost element comparison.

The performance of the RL-assisted model improves in stability and total cost with a higher number of training episodes and outperforms the AW policy. This is further investigated in the following. As shown in this figure, as we progress towards RL25000, the cost remains relatively stable with fewer observed outliers. To further analyze the performance of the RL-assissted model, the total cost in each episode is compared between the RL-assisted model using the Q-table after 25,000 episodes and the AW policy (Fig. 9). In this evaluation, the AR policy is excluded due to its poor performance as shown in Fig. 8.

In this figure, bars represent the total cost differences between the two policies for each testing episode. Blue bars indicate testing episodes where the RL-assisted model outperforms while yellow bars show testing episodes where the AW policy performs better. The chart is sorted in descending order, starting with cases where the RL-assisted model demonstrates better performance.

The chart reveals that the RL-assisted model outperforms the AW policy in 67 out of 100 testing episodes. Among the 33 episodes where the AW policy performs better, 21 show an insignificant total cost difference, with a gap of less than 1 %. This suggests that when the RL agent is exposed to a certain disruption profile, it generally makes good (comparable or better) decisions in response to disruptions. In these 100 sample cases, the cost savings by the RL-assisted model range from 0.11 % to 13.59 %. If the calculation only considers the affected requests, the highest savings from the RL-assisted model over the benchmark policy reaches 58,46 %. It should be noted that the amount of savings depends on the severity of delays caused by the disruptions in the network.

Further investigation at the action level reveals a correlation between the number of actions taken for a request and its total cost. This is especially valid for cases where a shipment is impacted by multiple disruptions. The Q-learning technique updates the action-value function for a given state-action pair by incorporating the immediate reward and the discounted future rewards, as described in Eq. 24. This approach captures the stochastic behavior of disruptions by updating the action-value function based on observed experiences. For states that are visited often, the action-value function undergoes repeated updates, eventually converging to an optimal value. For rarely encountered states, the action-value function remains less refined, as it is updated less frequently due to limited observations. In this example, such cases of multiple disruptions affecting one shipment are rare and therefore, the RL agent is not sufficiently exposed to such scenarios. Consequently, the action-value function suggests immediate actions without anticipating future disruptions, as the model has not frequently encountered these scenarios during training.

Additional investigations evaluate the relation between the decision made by the RL agent and individual cost components (i.e. storage, travel, handling, and delay). For the same cases, each cost component is compared between the two policies (Fig. 10). The y-axis scale is set to be the same with each tick representing 10,000 for an easier comparison across different cost components. Results show that the RL-assisted model tends to select options with higher travel cost (Fig. 10b), to reduce the storage cost and delay penalty as shown by the better performance of these two parameters in Fig. 10a and d. Meanwhile, handling costs are not significantly

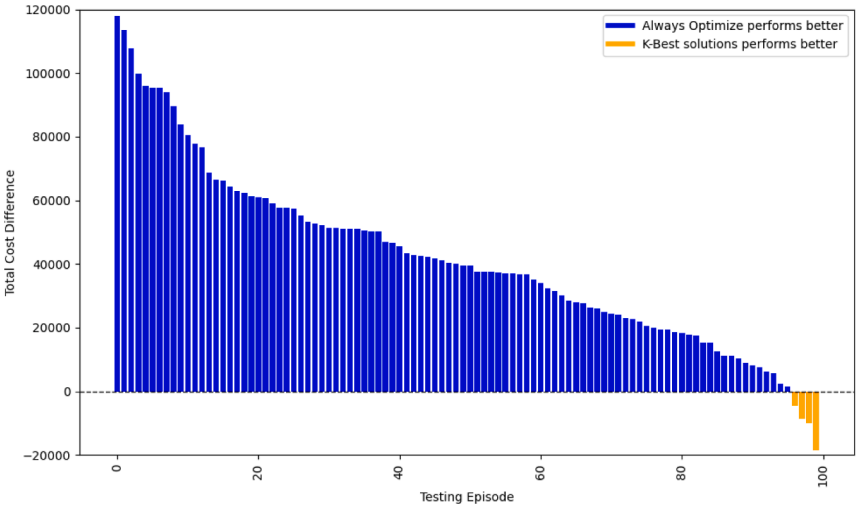



**Fig. 11.** Comparison between K-best solutions approach and always optimize in episodic simulations.

**Table 4**
Disruption scenarios.

| Profile | Severity | Location | Impact Type | LB Duration | UB Duration | Occurrence Rate (Lambda) | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | S1 | S2 | S3 | S4 | S5 |
| Profile1 | Low | Train | Delay | 3 | 6 | 0.00021 | 0.00042 | 0.00084 | 0.00168 | 0.00336 |
| Profile2 | Low | Barge | Delay | 3 | 9 | 0.00024 | 0.00048 | 0.00096 | 0.00192 | 0.00384 |
| Profile3 | High | Train, Barge | Delay | 12 | 48 | 0.00004 | 0.00008 | 0.00016 | 0.00032 | 0.00064 |
| Profile4 | Low | Terminal | Delay Capacity | 3 | 6 | 0.00021 | 0.00042 | 0.00084 | 0.00168 | 0.00336 |
| Profile5 | Low | Barge | reduction | 12 | 24 | 0.00004 | 0.00008 | 0.00016 | 0.00032 | 0.00064 |

affected by different policies. It should be noted that this 'behavior' is a result of the given cost parameters, directly linked to the reward system. This implies a different behavior if different cost parameters are considered during training.

In order to reduce the training computational load, the RL-assisted model selects an itinerary from the solution pool for reassignment (K-Best approach) and only triggers the optimization algorithm if no possible solution is found in the solution pool. By contrast, in an Always Optimize approach, the model activates the optimization module whenever a disruption occurs, generating optimal solutions considering all parameters in real time. This could result in different reassignment options for the same disruption scenario. To evaluate the gap between these two approaches, an experiment was conducted where the model, using an AR policy, compared solutions derived from the K-Best and Always Optimize methods. This allowed for a direct comparison of reassignment outcomes.

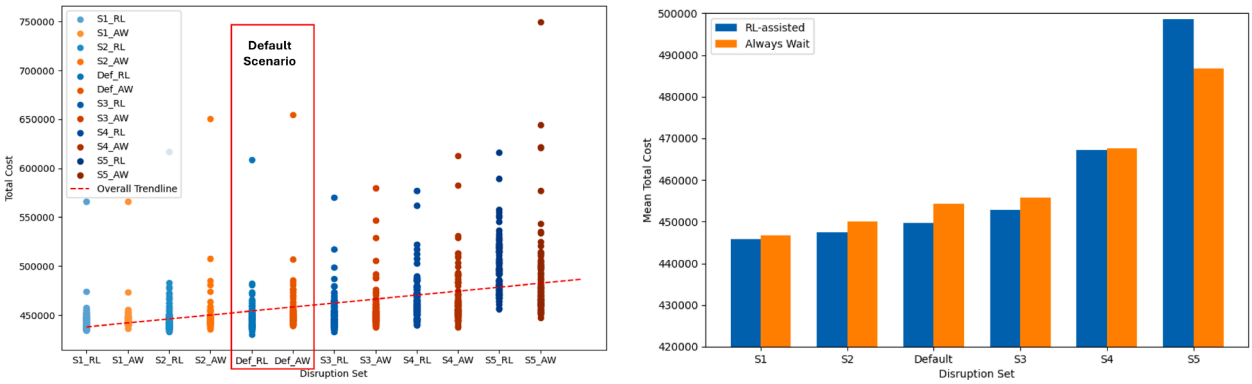
The results indicate that, in nearly all cases, solutions generated by the optimization module yielded lower total costs than those selected from the solution pool. The bars in Fig. 11 highlight the performance gap between the two approaches, with blue bars representing cases where the optimal solutions from Always Optimize approach were superior. This demonstrates an inherent optimality gap in the K-Best approach, suggesting opportunities for further improvement in the model. While the K-Best approach outperforms the benchmark policy, there remains potential to enhance reassignment solutions. This will be facilitated by utilizing efficient methods in terms of solution time, which enables the training process in a reasonable time.

However, it is crucial to recognize that utilizing optimization-derived solutions during the implementation phase after training with the K-Best approach poses challenges. If the reassignment option recommended by the optimization module for a given state has not been explored during training, the RL agent may select a suboptimal decision. For example, while the best decision might be waiting during a disruption, the RL agent could select the reassign option simply because it lacks prior experience with the optimal solutions.

### *5.3. Experiments with disruption frequencies*

We evaluate the behavior of the trained model in response to disruption scenarios with different frequencies compared to the ones encountered during training. This is done in comparison to the benchmark policy (AW).

Updated sets of disruption profiles are established and presented in Table 4. S1 is scenario 1 with the lowest occurrence rate. The occurrence rate keeps increasing two times for each disruption set until the set S5 with an extremely frequent occurrence rate. To put it in context, the occurrence rate of the default disruption set used in the training is between S2 and S3. The result of the second experiment is presented in Fig. 12a.




**(a)** Total Costs for Different Disruption Sets

**(b)** Aggregated Costs Comparison

**Fig. 12.** Experiment on different disruption sets results.

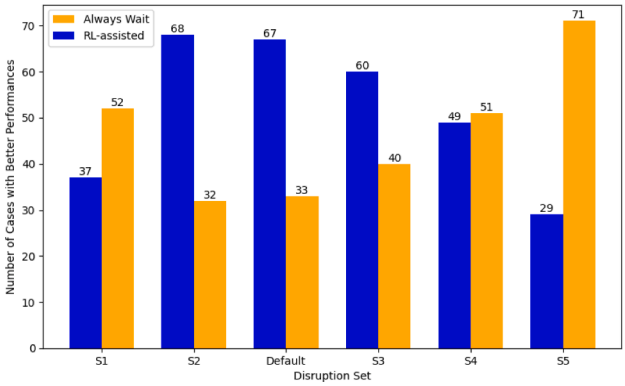



**Fig. 13.** Performance comparison by number of cases.

Using the same experimental setup, the model is simulated 100 times for each disruption set and policy. Fig. 12a shows the total cost of each simulation with the RL-assisted model represented by blue dots and the AW policy by orange dots. The first notable finding is that changes in the disruption occurrence rate affect the total costs, especially in S4 and S5. As disruption frequency increases, so do the total costs, as indicated by the ascending trendline.

An interesting observation is that, the RL-assisted model performs best under disruption scenarios S2 and S3. However, its performance declines compared to the benchmark policy as disruption frequency increases. Although the RL-assisted model shows more stable performance, indicated by fewer outliers, many cases yield higher costs compared to the AW policy as the disruption frequency increases. This decline in the performance is due to differences between the probability distributions in this experiment and those used during episodic training, e.g. S2 and S3 are most similar to the distributions used in training, allowing the RL-assisted model to perform well in these cases.

In the less frequent disruptions, the RL-assisted model shows a similar range of total costs to the benchmark policy. However, in the less frequent disruptions, the performance of the AW policy should be closer to the optimal policy, because the network is less disturbed. Additionally, the fact that the RL agent is triggered only around 10 times on average throughout 100 simulations, it reduces the chance of the RL-assisted to outperform the AW. In contrast, the RL agent is triggered almost 60 times on average in the default disruption set.

This finding is also indicated by the mean total costs comparison presented in Fig. 12b. The figure presents the total costs throughout the 20 simulations for each disruption set. It provides the overall performance of the policy. The chart shows that almost in all disruption sets, the mean total costs generated by the RL-assisted model are lower than the AW policy except in the S5. In particular, the mean total cost gap in S2 and S3 is larger compared to the other disruption sets.

Despite an overall better performance in the mean total costs, the data demonstrates that the RL-assisted model outperforms the benchmark policy only in scenarios S2 and S3 based on the number of simulations where it achieves better results. Specifically, Fig. 13 summarizes the performance across various disruption sets. A thorough investigation reveals there are several unstable action values in the learning agent, leading to suboptimal action selections, again due to the stochastic environment, suggesting the necessity of exposing the training process to larger disruption scenarios.

In disruption sets S2 and S3, the RL-assisted model performs better in 68 and 60 out of 100 simulations, respectively. This indicates that the RL-assisted model is more effective when the probability distribution closely resembles the episodic training. Meanwhile, in

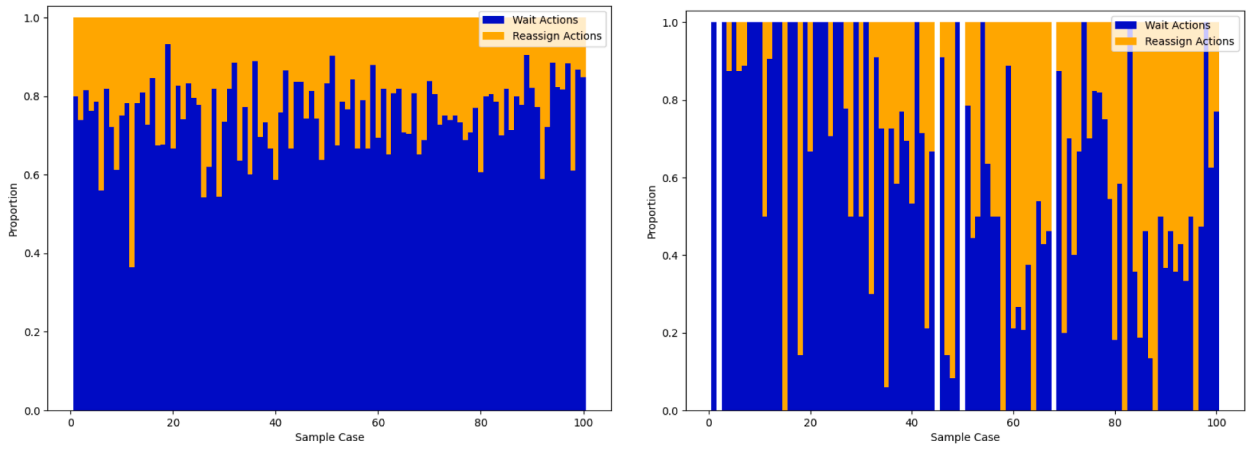



**Fig. 14.** Action proportion.

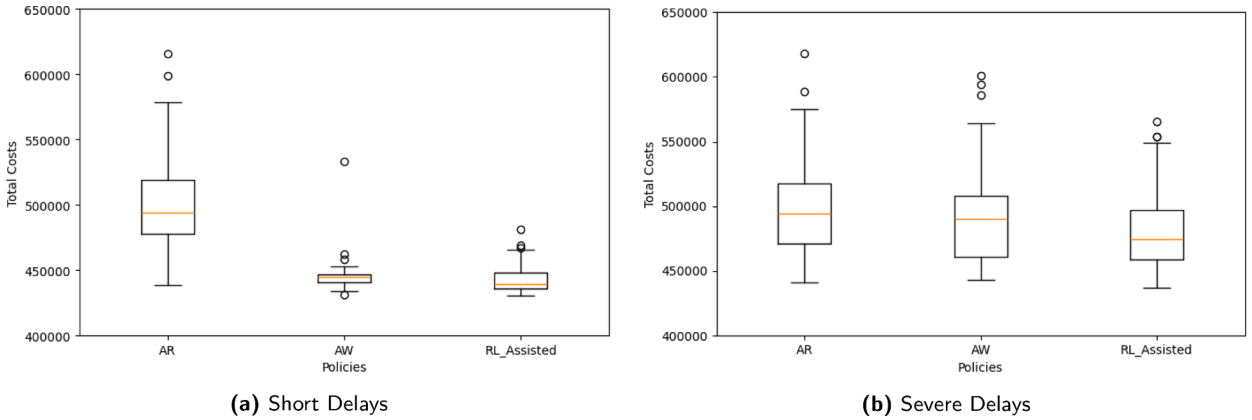



**Fig. 15.** Policy performance in different disruption severity.

disruption sets S4 and S5, the RL-assisted model performance is declining due to many disruptions in the network creating multiple disruptions for one shipment.

The next experiment on the disruption set involves simulating the model under two distinct scenarios: a moderate disruption scenario resulting in short delays and a severe disruption scenario causing severe delays. In the moderate disruption scenario, disruption profile 3 is excluded from the disruption set, while the severe disruption scenario includes only disruption profile 3. The occurrence rate used in this experiment is the same as the default case in the episodic training. However, to ensure a sufficient number of disruptions, the probability of occurrence is increased in the case of a severe disruption set. The results of this experiment are presented in Fig. 14, where the charts present the proportion of selected actions (Whether to wait or to reassign) by the RL agent. In moderately disrupted cases, decisions are dominated by choosing to 'wait', represented by the blue area, whereas this proportion shows a different pattern in case of severe disruptions. Blank cases demonstrate that no action is taken by the RL agent in that specific case, as the RL agent takes actions only in the case of affected shipments. The proportion of taking the reassign action (yellow area) is generally higher compared to the ones in a moderately delayed scenario.

Fig. 15 shows the policy performance in the two disruption settings. In the case of short delays, AW performs significantly better than AR, as waiting is generally the more effective action. However, under severe delay disruptions, the performance gap narrows, AR performs closer to AW, since reassigning becomes the better option in more cases. Overall, the RL agent demonstrates the ability to adapt its actions to different disruption settings. It learns to favor waiting in short delay scenarios and opts for reassignment more frequently when disruptions cause severe delays.

### *5.4. Comparison with benchmark reward policy*

The proposed model is further analyzed by benchmarking it against an alternative reward policy inspired by Zhang et al. (2023). Their model provides a solution for shipment matching decisions in response to disruptions using a binary reward policy, i.e., the RL

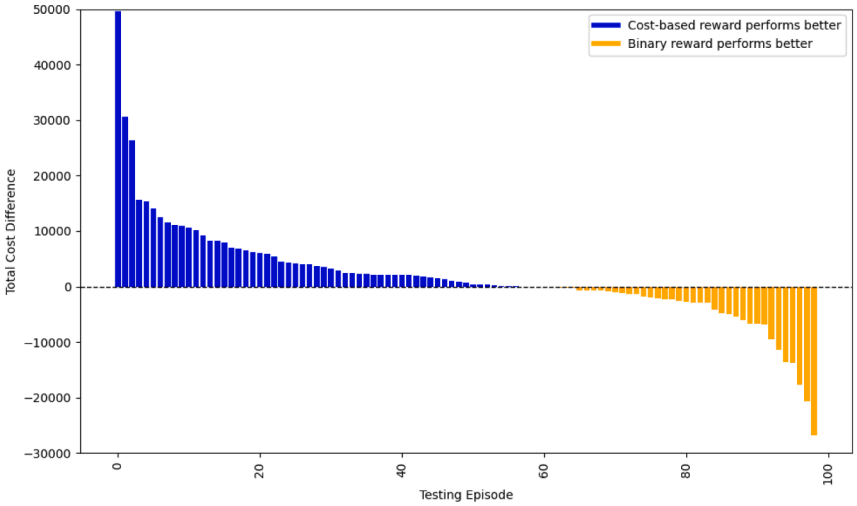



**Fig. 16.** Comparison between different reward policy.

agent receives a reward of 0 or 1 depending on whether the selected action creates delays in shipment delivery. Available actions include waiting and removal-insertion at each visited terminal. In Zhang et al. (2023), the reward is determined by the delay of the entire itinerary, whereas our study adopts a service line-specific approach for reward calculation.

Additionally, the removal-insertion action can yield up to 2 reward points. The first point is awarded for the removal action if it successfully avoids a delay in the shipment's original itinerary. To determine this, the model estimates the actual arrival time after the disruption ends and compares it with the shipment's due time. An additional point is granted if the insertion action chooses an alternative itinerary that does not result in a delay at its conclusion. However, even if the alternative itinerary introduces a delay at the end, the RL agent still earns 1 reward point for avoiding a delay in the original itinerary.

Considering the implementation of a rolling horizon approach by Zhang et al. (2023) and to create the same simulation environment for benchmarking, the following modifications are made to the object-oriented discrete event simulation framework of this study:

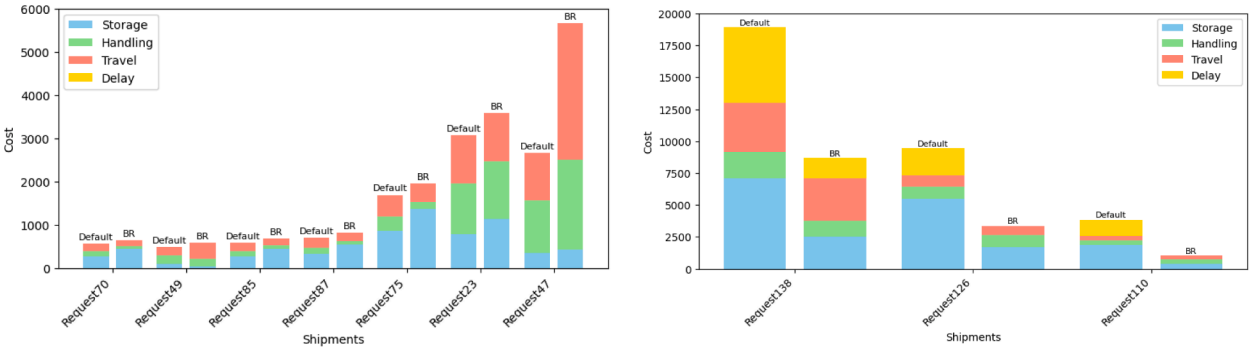
1. In our proposed reward system, the reward is given for an action that consists of one service line instead of one itinerary. We have adapted the mechanism to the delivery reward mechanism, where a reward is only given for an action that delivers the shipment to the end destination depending on whether a delay occurs or not. For an action that carries the shipment to a transshipment terminal, the reward is set to 0.
2. The reward for the reassign action can only generate 1 point. Even if it avoids a potential delay by taking the alternative itinerary, it still gets a reward of 0 if the alternative itinerary also creates a delay at the end.

The experiment is performed using the same setting as with the AW policy. Both approaches are simulated for 100 testing episodes and the total costs of each episode are compared and results are presented in Fig. 16. Results show that, under the simulation settings, the cost-based reward approach outperforms the benchmark approach in 59 out of 100 cases, caused by the fundamental difference between the reward policy of the two approaches.

The cost-based reward incorporates additional costs including storage and travel costs, in addition to the delay penalties. In contrast, the benchmark policy considers only the delay penalties. As a result, the RL model using a binary reward often selects actions that, while minimizing delays, may incur higher travel and storage costs that outweigh any delay penalty reductions, particularly in cases of minor delays. This leads to a higher total cost compared to the cost-based reward approach, which balances all cost factors more effectively.

To gain a clearer understanding of how these two reward policies perform, further investigation is conducted. Cases with significant differences in total costs are analyzed at the shipment level, focusing on instances where the cost-based reward outperforms and where the binary reward shows better performance in terms of total costs. Among 200 shipments considered in the simulation, the investigation considers those with a significant cost difference between the two reward policies. A sample comparison for each case is presented in Fig. 17.

Fig. 17a illustrates a scenario where the cost-based reward (default scenario) performs better, while Fig. 17b shows a case favoring the binary reward approach. The stacked bar charts display the proportion of each cost component (storage, travel, handling, and delay), with delay costs highlighted in yellow. The left chart shows minimal delay costs compared to the right chart, suggesting that the cost-based reward is more effective when disruptions do not result in extended delays. In contrast, the binary reward policy is more effective when disruptions incur significant delay penalties, aligning with the mechanism of each reward type. The binary reward policy focuses on avoiding delays, thus making RL agent decisions less impactful when disruptions cause only minor delays. This is further validated by exposing the system to disruptions of higher frequency severity (compared to the experiment described




**(a)** Cost-based (Default) Reward Performs Better

**(b)** Binary Reward (BR) Performs Better

**Fig. 17.** Shipment level comparison for reward policy performance analysis.

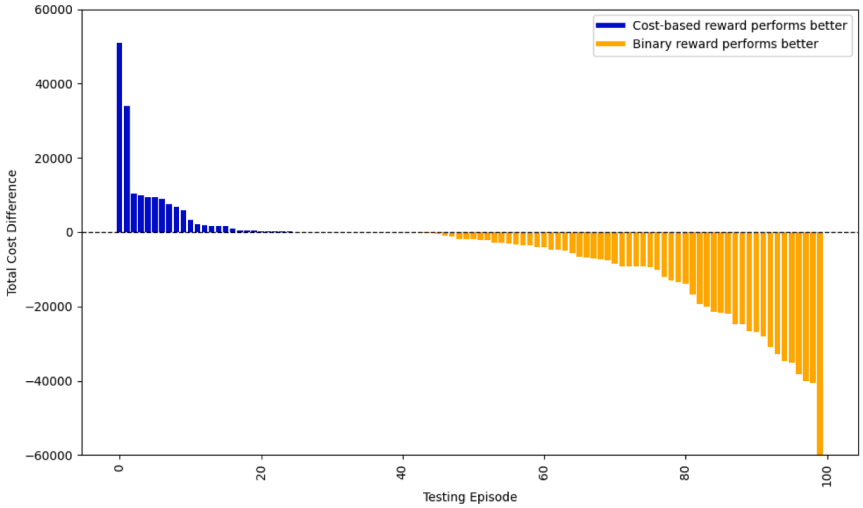



**Fig. 18.** Comparison in severe disruptions set.

in Section 5.3). As expected, the network experienced generally higher delays due to these severe disruptions. The results, presented in Fig. 18, show that under these conditions, the binary reward policy outperforms the cost-based reward policy.

In practical applications, selecting the appropriate reward policy could be tailored to the disruption characteristics, enabling the creation of 'action cards' to respond effectively to disruptions. For networks where disruptions typically cause minor delays, a cost-based reward system might be more advantageous. In contrast, networks prone to significant delays, or those involving delays-sensitive shipments such as high-value cargo-may benefit from a binary reward policy. While the cost-based reward demonstrates better overall performance by balancing multiple cost factors, its effectiveness relies on the cost parameters used during training, rendering it more sensitive to cost variations. If actual costs deviate from those in training, the value function might become less effective, while this should not be the case in the binary reward approach. However, considering that costs tend to be relatively stable in the typical application scenarios, this deviation is unlikely to substantially affect performance. Meanwhile, the binary reward approach remains a robust alternative for scenarios where cost parameters are less predictable.

It is essential to recognize that the binary reward adaptation in this study does not fully capture the performance characteristics of the model presented in Zhang et al. (2023), as several elements differ from the original setup. For instance, the referenced model employs an ALNS for optimization and integrates a deep learning approach for the learning agent. Nonetheless, by isolating the comparison to the reward system alone, this analysis still provides valuable insights into the relative performance of these two distinct reward policies.

## 6. Conclusions and recommendations for future research

This paper proposes a modular simulation-optimization decision support tool to address the dynamic nature of demand and disruptions in synchromodal transport. To address the stochastic nature of disruptions in the synchromodal framework and enable efficient disruption management under unpredicted events, aRL agent is integrated that can select actions between *wait*- where a shipment always stays with its original itinerary even though disruptions occur- and *reassign*- where a shipment is reassigned to a

new itinerary in reaction to disruptions. By leveraging its 'experience' to make informed decisions, the RL-assisted model outperforms the benchmark policy (*always wait*) in choosing the proper actions in response to disruptions. Testing the approach on a real case shows that the RL-assisted model outperforms the benchmark policy in many cases by achieving significant cost savings considering affected shipments. Analysis of the overall performance suggests that the proposed model has the potential to create a more resilient synchromodal framework by utilizing reinforcement learning.

This paper integrates simulation and optimization, leveraging simulations to enable scalability and consideration of dynamics in large, complex systems prone to various uncertainties. By easily adjusting parameters such as service time distributions, delays, and disruptions, simulations facilitate the exploration of diverse scenarios and the stress-testing of solutions. Furthermore, simulations allow RL models to account for rare but critical events, such as extreme delays, which are often difficult to observe in limited real-world data but can significantly impact system performance.

Regardless of the extent of cost reduction, which depends on network settings, the model is able to perform well across various disruption scenarios, achieving its design objectives.

The comparison of different reward policies indicates a better performance of the proposed model (cost-based reward) compared to a binary (1/0) rewarding system, given the same simulation settings. The cost-based reward considers a more comprehensive trade-off of all cost components as opposed to the binary reward system which focuses on avoiding delays resulting in lower overall total costs. However, each approach exhibits strengths depending on the disruption severity: the cost-based reward performs better in scenarios with minor delays, whereas the binary reward system is more effective when disruptions result in prolonged delays. From an application perspective, this research offers valuable insights for stakeholders operating in hinterland freight transport through an advanced decision support system. It offers shippers a better understanding of cost trade-offs within the network, enabling them to make more informed decisions. This system is particularly beneficial as shippers can be highly sensitive to the impact of disruptions on their systems. The framework enables the simulation of 'hypothetical' disruption scenarios and comparing the outcome with those of experts.

The results of this research indicate that utilizing a learning approach within a synchromodal framework could bring added value serving as a foundation for future research directions. Firstly, incorporating data-driven disruptions and extensive profiles, such as severe disruptions at terminals or service line infrastructure, including the possible correlation between different disruption profiles, could provide more realistic scenarios for the RL agent and enhance the reliability of the model. In addition, incorporating cascading effects of disruptions and how they propagate through the network, will provide a more realistic approach to modeling disruptions. Introducing these dependencies between the disruption profiles will allow the RL agent to learn more robust mitigation strategies, improving its overall performance. While the disruption profiling in this paper was designed to balance the training efficiency and real-world complexity, further exploration into profile configurations, identifying optimal settings for training performance, could enhance the effectiveness of the proposed framework. Secondly, incorporating a more advanced optimization model and solution approach allows for the full utilization of optimization and simulation, ensuring that the optimization module is consistently activated during training. Thirdly, incorporating more advanced learning techniques will increase the robustness, reliability, and capability of the modeling approach. Finally, the comparison of cost-based and binary reward policies provides insight into the advantages of each approach in different situations. Another direction for future research is the development of a robust disruption management system that combines these reward policies, leveraging the strength of each reward policy. Finally, future work could explore the application of our approach to transportation networks in other contexts to explore its generalization ability and integrate real-time event data to further validate and refine its performance.

## CRediT authorship contribution statement

**Satrya Dewantara:** Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Conceptualization; **Siyavash Filom:** Writing – review & editing, Validation, Software, Methodology, Investigation; **Saiedeh Razavi:** Writing – review & editing, Validation, Investigation; **Bilge Atasoy:** Writing – review & editing, Validation, Supervision, Methodology, Formal analysis; **Yimeng Zhang:** Writing – review & editing, Validation; **Mahnam Saeednia:** Writing – review & editing, Validation, Supervision, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Conceptualization.

## Acknowledgments

We would like to acknowledge and appreciate AI Port Center (convergence.nl) for supporting the Catalyzer project SynCo2Log, Synchromodality for Decarbonization and Logistics Optimization.

## Appendix A.

The network and service schedule datasets are adapted from Zhang et al. (2023), while the demand data is synthetically generated. These datasets are available on the research website[1]. Cost parameters, as detailed in Table A.5, are derived from Zhang et al. (2022).

[1] https://figshare.com/articles/dataset/Datasets_for_Learning_Assisted_Hybrid_Simulation_Optimization_Model/28551299

**Table A.5**
Cost parameters.

| Cost Parameters | Unit | Barge | Train | Truck |
|---|---|---|---|---|
| **Mode related cost** | | | | |
| Travel cost (time) | Euro/TEU/hour | 0.6122 | 7.54 | 30.98 |
| Travel cost (distance) | Euro/TEU/km | 0.0213 | 0.0635 | 0.2758 |
| Handling cost | Euro/TEU | 3 | 18 | 18 |
| **Shipment related cost** | | | | |
| Storage cost | Euro/TEU/hour | 1 | 1 | 1 |
| Delay penalty | Euro/TEU/hour | 1 | 1 | 1 |

## References


Ambra, T., Caris, A., Macharis, C., 2019. Should I stay or should I go? Assessing intermodal and synchromodal resilience from a decentralized perspective. Sustain. (Switzerl.) 11 (6). https://doi.org/10.3390/su11061765

Barkley, A., Mcleod, K., 2022. Congestion and consolidation: an empirical study of a barge shipping merger. Reg. Sci. Urban Econ. 93. https://doi.org/10.1016/j.regsciurbeco.2021.103725

CCNR, 2020. Water levels and available vessels' draught at Gauging stations on Rhine and Danube https://inland-navigation-market.org/chapitre/3-water-levels-and-freight-rates-2/?lang=en.

Chen, L., Miller-Hooks, E., 2012. Resilience: an indicator of recovery capability in intermodal freight transport. Transport. Sci. 46 (1), 109–123. https://doi.org/10.1287/trsc.1110.0376

Delbart, T., Molenbruch, Y., Braekers, K., Caris, A., 2021. Uncertainty in intermodal and synchromodal transport: review and future research directions. https://doi.org/10.3390/su13073980

Di Febbraro, A., Sacco, N., Saeednia, M., 2016. An agent-based framework for cooperative planning of intermodal freight transport chains. Transport. Res. Part C: Emerg. Technolog. 64, 72–85. https://doi.org/10.1016/j.trc.2015.12.014

van Dorsser, C., Vinke, F., Hekkenberg, R., van Koningsveld, M., 2020. The effect of low water on loading capacity of inland ships. Eur. J. Transp. Infrastruct. Res. 20 (3), 47–70. https://doi.org/10.18757/ejtir.2020.20.3.3981

Filom, S., Razavi, S., 2025. A learning-based robust optimization framework for synchromodal freight transportation under uncertainty. Transport. Res. Part E: Logist. Transport. Rev. 195, 103967.

Guo, W., Atasoy, B., Beelaerts van Blokland, W., Negenborn, R.R., 2020. Dynamic and stochastic shipment matching problem in multimodal transportation. Transp. Res. Rec. 2674 (2), 262–273. https://doi.org/10.1177/0361198120905592

Guo, W., Atasoy, B., van Blokland, W.B., Negenborn, R.R., 2021. Global synchromodal transport with dynamic and stochastic shipment matching. Transport. Res. Part E: Logist. Transport. Rev. 152. https://doi.org/10.1016/j.tre.2021.102404

Guo, W., Atasoy, B., Negenborn, R.R., 2022. Global synchromodal shipment matching problem with dynamic and stochastic travel times: a reinforcement learning approach. Ann. Oper. Res. https://doi.org/10.1007/s10479-021-04489-z

Hrušovský, M., Demir, E., Jammernegg, W., Van Woensel, T., 2021. Real-time disruption management approach for intermodal freight transportation. J. Clean. Prod. 280. https://doi.org/10.1016/j.jclepro.2020.124826

Jaimungal, S., 2022. Reinforcement learning and stochastic optimisation. Finance Stochast. 26 (1), 103–129.

Memarian, B., Doleck, T., 2024. A scoping review of reinforcement learning in education. Comput. Educ. Open 6, 100175. https://doi.org/10.1016/j.caeo.2024.100175

Palmqvist, C.W., Lind, A., Ahlqvist, V., 2022. How and why freight trains deviate from the timetable: evidence from Sweden. IEEE Open J. Intell. Transport. Syst. 3, 210–221. https://doi.org/10.1109/OJITS.2022.3160546

PoR, 2024. Barge Performance Monitor. https://www.portofrotterdam.com/en/logistics/connections/intermodal-transportation/inland-shipping/barge-performance.

Qu, W., Rezaei, J., Maknoon, Y., Tavasszy, L., 2019. Hinterland freight transportation replanning model under the framework of synchromodality. Transport. Res. Part E: Logist. Transport. Rev. 131, 308–328. https://doi.org/10.1016/j.tre.2019.09.014

Rodríguez-Clare, A., Ulate, M., Vasquez, J.P., 2023. Supply Chain Disruptions, Trade Costs, and Labor Markets A Framework of International Trade with Unemployment. Technical Report.

Schlake, B.W., Barkan, C. P.L., Edwards, J.R., 2011. Train delay and economic impact of in-service failures of railroad rolling stock. Transp. Res. Rec. (2261), 124–133. https://doi.org/10.3141/2261-14

Statista, 2022. Average monthly delays for late container vessel arrivals worldwide from January 2019 to July 2022 https://www.statista.com/statistics/1303383/average-delays-for-late-ship-arrivals-worldwide/.

Sutton, R.S., Barto, A.G., 2018. Reinforcement Learning : an Introduction, Cambridge: MIT press.

Tavasszy, L.A., Behdani, B., Konings, R., 2017. Intermodality and synchromodality. 251–266. SSRN Electronic Journal. https://doi.org/10.2139/ssrn.2592888

Wide, P., Kalahasthi, L.K., Roso, V., 2022. Efficiency effects of information on operational disruption management in port hinterland freight transport: simulation of a Swedish dry port case. Int. J. Logist. Res. Applic. https://doi.org/10.1080/13675567.2022.2100333

Zhang, Y., Guo, W., Negenborn, R.R., Atasoy, B., 2022. Synchromodal transport planning with flexible services: mathematical model and heuristic algorithm. Transport. Res. Part C: Emerg. Technolog. 140. https://doi.org/10.1016/j.trc.2022.103711

Zhang, Y., Negenborn, R.R., Atasoy, B., 2023. Synchromodal freight transport re-planning under service time uncertainty: an online model-assisted reinforcement learning. Transport. Res. Part C: Emerg. Technolog. 156. https://doi.org/10.1016/j.trc.2023.104355