An Optimisation–Simulation Framework for Dynamic Fleet Re-optimisation in Autonomous Pod-Based Rail Freight

Aditya Pavadad
Msc. Transportation , Infrastructure and Logistics
Delft University of Technology
Delft, The Netherlands

***Abstract*—This paper develops a hybrid decision-making framework for autonomous pod-based railway systems, integrating Mixed-Integer Linear Programming (MILP) with a Discrete-Event Simulation (DES) to manage dynamic and uncertain operating conditions. Initial schedules are optimised using MILP and then executed in DES, with re-optimisation triggered by disruptions such as delays, carrier breakdowns, or network failures. This event-driven feedback loop enables continuous adjustment of plans, bridging the gap between static planning and real-time operations. Validation on a toy case and the Randstad network shows that dynamic re-optimisation significantly improves fulfillment rates and carrier utilisation while mitigating the fragility of static schedules. The analysis reveals how resilience is achieved through flexible delivery windows, resource reallocation, and network redundancy, though often at the cost of repositioning and delays. The results provide both theoretical and practical insights, emphasising that digital re-planning capabilities, fleet pre-positioning, and flexible service standards are essential for building robust autonomous freight systems.**

***Index Terms*—Autonomous rail systems, discrete-event simulation (DES), mixed-integer linear programming (MILP), disruption management, resilience, freight logistics.**

I. Introduction

Intermodal transportation systems play a pivotal role in enhancing the efficiency and sustainability of modern freight distribution. By integrating different transport modes, these systems can reduce transportation costs and CO_2 emissions, addressing environmental concerns while maintaining economic viability. A key area of innovation is the improvement of rail freight, which has faced challenges in flexibility and reliability compared to road transport. One such innovation is the development of autonomous wagons, or "pods," which aim to significantly enhance the flexibility of rail operations through a modular vehicle architecture. A pod consists of a transport unit (TU) and a carrier unit (CU), as shown in image 1. These systems, central to projects like Pods4Rail [1], enable individual carriers to join or detach from platoons, optimising both resource utilisation and network capacity.

The core challenge in operating such a system lies in the optimal assignment of carriers to transport units (TUs) and the strategic repositioning of empty carriers to serve future demand. Static optimisation methods, such as Mixed-Integer Linear Programming (MILP), can generate highly efficient

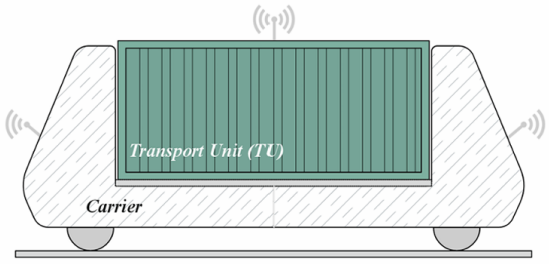

Fig. 1. Autonomous Pod Rail: TU on Carrier Unit

baseline schedules under deterministic conditions. However, these static plans are inherently fragile. They can quickly become infeasible when faced with real-world operational disruptions, including carrier breakdowns, network link closures, or unforeseen variations in demand [2]. The inability of static models to adapt to dynamic events necessitates a more resilient planning approach.

To address this, this paper proposes and evaluates a hybrid simulation-optimisation framework that integrates MILP with Discrete-Event Simulation (DES) to enhance disruption management in autonomous rail systems. The methodology involves a concise process: conducting a targeted literature review to establish a foundation, generating synthetic data to reflect operational scenarios, and identifying key factors—such as temporal, spatial, and operational constraints—affecting carrier-TU assignments. A DES model is developed using SimPy to simulate dynamic system behaviour, capturing events like pod arrivals and carrier assignments. In contrast, an MILP model optimises initial schedules and re-optimises them in response to disruptions. Performance is assessed via KPIs including efficiency, service reliability, and utilisation, with results systematically documented to provide actionable insights [3]–[5]. This approach operates on an event-driven feedback loop: the MILP generates an initial optimal plan executed in the DES environment, pausing upon disruption to capture the real-time system state, which informs a new adaptive plan.

The main contribution of this work is the development and validation of this simulation-optimisation framework as a practical tool for disruption management in autonomous rail

systems. By testing the system against a range of controlled disruptions in both a small-scale and a large-scale network, this research provides quantitative insights into system resilience, identifies key operational trade-offs, and demonstrates the critical role of service flexibility in maintaining performance under uncertainty. This study bridges the gap between static planning and dynamic operational execution, offering a robust methodology for designing and managing the next generation of autonomous freight logistics networks.

II. Literature Review

Within the broader context of the Pods4Rail [6] project, significant advancements have been made in rail-based, intermodal freight transport systems. Liao, Han, and Saeednia [4] have explored ways to enhance system flexibility through modular vehicle routing, focusing on the integration of autonomous wagons in railway environments. Their research addresses key operational challenges like platooning and routing efficiency, demonstrating potential reductions in transportation costs and improvements in capacity utilisation [7].

A. Carrier-TU Matching and Assignment

The optimisation of resource allocation in transportation systems has been widely studied, particularly in ride-sharing, autonomous vehicles, and railway logistics. While limited research addresses autonomous pod railway systems, existing studies provide valuable insights into key factors, optimisation methods, and evaluation approaches that can be adapted to this domain.

Tafreshian et al. (2020) [8] conducted a comprehensive review of ride-matching algorithms in peer-to-peer (P2P) rideshare systems, classifying them as one-to-one, one-to-many, and many-to-many matching. This discussion of temporal and spatial constraints aligns closely with challenges in autonomous pod railway systems, where carriers must navigate fixed railway networks and adhere to strict time windows.

Wu et al. (2008) [9] presented decentralised P2P shared ride systems with explicit spatial constraints and geospatial matching of riders and drivers. Ma et al. (2019) [10] introduced heuristic algorithms to solve P2P ridesharing matching problems using recursive techniques based on feasibility constraints and preference lists.

Bei and Zhang (2018) [11] developed a two-phase algorithm using minimum weight perfect matchings that guarantees no vehicle bears more than one request. This approach minimises travel distances and operational expenses, providing a solution framework adaptable to rail carrier-TU assignments.

B. Empty Carrier Relocation Strategies

Efficient relocation strategies are crucial for maintaining adequate carrier availability while reducing operational costs. In railway operations, Alfieri et al. [12] employed integer multicommodity flow models with transition graphs to optimise train unit circulation. Peeters and Kroon [13] enhanced this approach using branch-and-price methods for efficient train unit allocation across lines.

For car-sharing systems, Illgen and Höck [14] conducted a systematic review of Vehicle Relocation Problems (VReP), supporting the effectiveness of mixed-integer programming and multistage methods for relocation management. Clemente et al. [15] modelled user-based relocation strategies using Timed Petri Nets (TPN), focusing on load balancing in car-sharing systems.

Weikl, Bogenberger & Cepolina et al. (2012) [16], [17] investigated operator-based relocation methods and two-step algorithms combining predictive offline planning with adaptive real-time adjustments. Ait-Ouahmed et al. [18] analysed greedy, Tabu, and Iterated Local Search algorithms for optimal vehicle distribution strategies.

C. Literature Gaps and Research Contributions

Eder et al. (2025) study a static, passenger-oriented POD routing model and MILP/ALNS solution—optimising customer arrivals and operator costs on Munich Zone M ($\approx$17% faster arrivals; $\approx$21% less base travel). In contrast, our framework targets freight operations and is dynamic and disruption-aware, coupling MILP with DES to re-optimise with lock-ins, manage empty-carrier flows, and model platooning on rail-pod systems [19].

While previous studies on ride-sharing, car-sharing, and railway logistics have made valuable contributions, several gaps remain. The absence of autonomous pod-based research stems from the focus on existing road-based systems or traditional railway logistics, raising novel issues like carrier-TU assignment in hybrid road-rail scenarios.

Current relocation strategies are largely user-incentive driven, which do not work for systems that move Transport Units (TUs). This project focuses on developing system-driven relocation strategies that do not rely on user incentives, ensuring efficient resource utilisation and high service reliability.

Moreover, while platooning effects are well-studied for road-based systems, they are nearly non-existent for modular pod systems. The developed models incorporate platooning to guarantee optimal carrier availability and scheduling when multiple pods move simultaneously.

Finally, while Rolling Horizon approaches are not novel, their application to carrier-TU assignment and relocation in flexible modular pod rail systems represents a new contribution. This study implements Rolling Horizon as a primary tool, integrated with MILP and DES, providing a robust framework for dynamic decision-making and system evaluation under various scenarios.

To clarify the relationship between related problem components and autonomous pod-based rail system elements, Table I provides a mapping of these components.

III. System Modeling & Hybrid Framework

This section details the underlying model of the autonomous pod-based railway system and the integrated simulation-optimisation framework developed for its analysis. The system is conceptualised as a Discrete Event System (DES), a modelling approach well-suited for capturing the dynamics of

TABLE I
MAPPING COMPONENTS TO POD CONTEXT

Component in Literature	Equivalent in Pod Context
Car/Ride-Sharing	
Car/Vehicle	Carrier
User/Passenger	Transport Unit (TU)
Station/Pickup-Drop-off	Station
Railway Logistics	
Rolling Stock (Train Unit)	Carrier
Passenger/Goods	Transport Unit (TU)
Station/Depot	Station
Empty Rolling Stock	
Empty Train Units	Empty Carriers

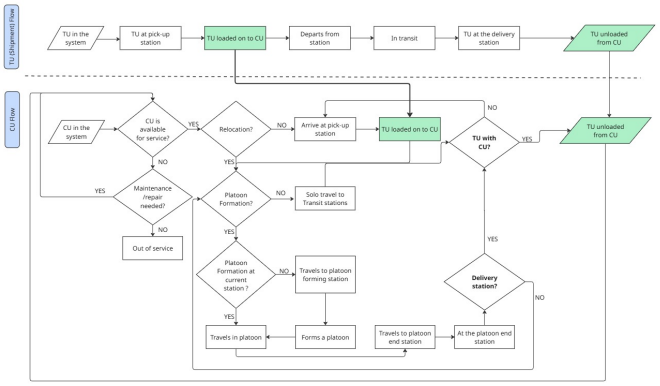

Fig. 2. Simulation flow process with platooning

shared mobility networks [2]. In a DES, the system's state changes only at discrete points in time in response to specific events.

A. Overview of DES Logic

The algorithm structure of the Discrete Event Simulation is shown in Algorithm 1. The DES initiates parallel processes for all transport units (TUs) and carrier units (CUs). Each TU follows a linear sequence: arrival at pickup, loading, loaded departure, travel to delivery, and unloading. Concurrently, CUs advance arc by arc in their itineraries, checking for assigned TUs and platoon membership before departure (loaded/empty, solo/platoon). Travel triggers intermediate events, with arrivals logging deliveries and unloads at drop-offs or repositioning at stations. The simulation advances event-by-event until completion, yielding a comprehensive event trace.

B. System Dynamics and State-Event Model

The core entities of the system are the **Transport Units (TUs)**, which represent the freight or cargo capsules, and the **Carrier Units (CUs)**, which are the autonomous, self-propelled wagons that transport the TUs. The operational lifecycle of a transport request involves a TU arriving at a pickup station, waiting for an available CU, being loaded, and then being transported to its destination. During transit, CUs may travel solo or form platoons to enhance efficiency before the TU is ultimately unloaded at its destination station, completing the request.

When a transport request (TU) enters the system, it first arrives at its designated pick-up station and remains there until a carrier unit (CU) becomes available. The CU then drives empty to that station (if it is not already positioned), waits for the TU to be ready, and initiates the loading operation. Once loading completes, the combined CU–TU pair departs immediately and traverses the network along the predefined route toward the delivery station. Travel occurs without interruption until the pair reaches the drop-off point, where the TU remains on board while unloading is performed. Upon completion of unloading, the TU exits the system, and the CU departs the station empty. The empty CU may then either proceed to the next pick-up station, reposition itself in anticipation of future requests, or enter maintenance if required. This basic sequence—arrival, wait, load, in-transit movement, unload, and empty departure and captures the baseline operational logic without any coordinated grouping (platooning) of CUs.

Figure 2 extends the baseline sequence by allowing carrier units to travel in platoons both when empty and when carrying a transport unit. After completing a loading or unloading operation, each CU checks whether it should form or join a platoon before its next departure. If platooning is desired (platoon stations are predetermined and known from the MILP output), a CU will either wait at its current station for other CUs to arrive or move to a predefined station. Once at least two CUs converge—whether they are empty or loaded—they merge into a single platoon, synchronise their departure time and speed, and traverse the network together along the same route. The platoon remains intact until individual CUs reach a station where one of two events occurs: (1) a loaded CU arrives at its delivery station and unloads its TU, or (2) an empty CU arrives at a designated split station (either to pick up a TU or to further form a platoon). At that moment, the group dissolves: unloaded CUs become available for new tasks or maintenance, and loaded CUs (if still in transit) continue to their final delivery station alone. By enabling both empty and loaded platooning, this enhancement preserves the core steps of arrival, loading, transit, unloading, and empty movement while improving resource utilisation and travel efficiency.

To formally capture these dynamics, a state-event model is defined for both entities. A state describes the operational condition of a unit at any given moment (e.g., waiting, in-transit), while an event is an instantaneous occurrence that triggers a transition between states (e.g., arrival, loading completion). The key states for TUs and CUs are summarised in Table II and Table III, respectively.

To formally capture these dynamics, a state–event model is defined for both entities. As illustrated in Figure 3, **CU** behaviour is non-linear: carriers can loop between staging, loading/unloading, en-route travel, and optional platoon join/leave, depending on the triggering events (E1–E10). In contrast, the **TU** lifecycle in Figure 4 follows a linear pipeline—from arrival to loading, transit, and unloading—which completes the request. The state codes (C0–C8, T0–T4) correspond to

the definitions in Table III and Table II; arrows are labelled by the event that causes the transition (e.g., unloading E9; empty repositioning E10).

TABLE II
KEY STATES FOR A TRANSPORT UNIT (TU)

State Code	Description
T0	At Station, waiting for a CU.
T1	Loaded onto a CU, ready for departure.
T2	In Transit towards its destination.
T3	Arrived at delivery station, awaiting unload.
T4	Delivered; journey complete.

TABLE III
KEY STATES FOR A CARRIER UNIT (CU)

State Code	Description
C0	Idle and unassigned.
C2	In Transit with a TU (Solo).
C3	In Transit with a TU (Platoon).
C5	At Station, waiting for a task.
C6	Repositioning to another station (Empty, Solo).
C8	Out of Service (e.g., maintenance).

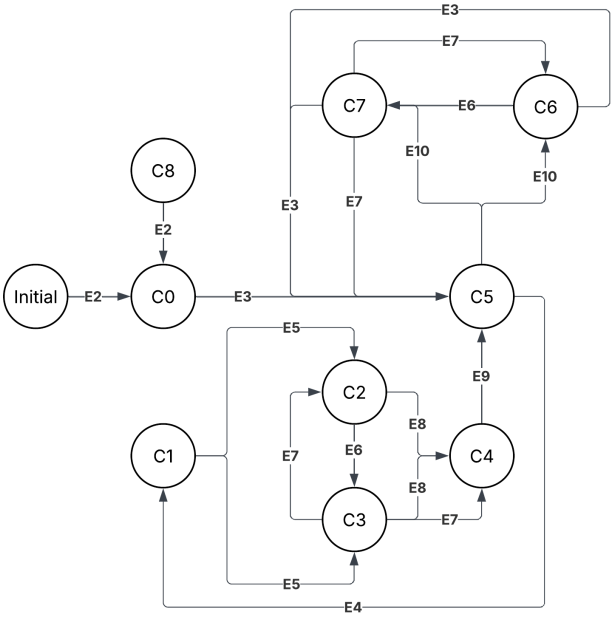

Fig. 3. Carrier Unit (CU) Event-State Transition Diagram

C. The Integrated DES-MILP Framework

While static optimisation can produce an optimal schedule, it cannot respond to unforeseen events. To address this, a hybrid framework that couples Mixed-Integer Linear Programming (MILP) with DES was developed to enable dynamic re-planning. The MILP acts as the decision-making core, formulating carrier assignment, routing, and relocation as an

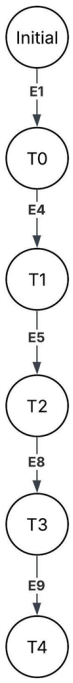

Fig. 4. Transport Unit (TU) Event-State Transition Diagram

optimisation problem under temporal and spatial constraints. Its outputs provide a structured schedule of transport and carrier movements, which are then translated into simulation-ready inputs for the DES to capture system dynamics and operational uncertainties [20]. This framework operates on an event-driven feedback loop, as summarised in Algorithm 2 and detailed below:

1) **Initial Plan:** An MILP solver first generates a globally optimal baseline plan based on all known transport requests, carrier availability, and network topology. This plan dictates the precise assignment, timing, and routing for all CUs.
2) **Simulation:** The DES environment executes this static plan, simulating the movement of CUs and TUs through the network event by event.
3) **Disruption Trigger:** During the simulation, a disruption is introduced (e.g., a CU breakdown, a new TU request, or an arc closure). This event invalidates the current optimal plan.
4) **System Snapshot and Lock-in:** The DES immediately pauses and generates a ”snapshot” of the current system state. This includes the precise location and status of every CU and TU. Any operations already in progress are ”locked in” and allowed to complete to ensure operational continuity.
5) **Re-Optimisation :** The system snapshot is fed back as the initial condition to the MILP solver. The MILP then re-optimises the remaining unassigned tasks and new requests to generate a new, resilient recovery plan for

the rest of the time horizon.

6) **Continuation:** The DES resumes the simulation, executing the new, adapted plan until the next disruption occurs or the simulation ends.

This iterative process provides the flexibility required to adapt dynamically to uncertainty and perform on-demand re-optimisation, bridging the gap between static planning and real-time operational execution.

IV. Experimental Setup

To evaluate the performance and resilience of the proposed framework, a series of simulation experiments was conducted. This section details the quantitative performance metrics used for evaluation and describes the two distinct case studies designed to test the system under both controlled and complex operational conditions.

A. Key Performance Indicators (KPIs)

We assess performance using three KPI groups: service effectiveness, resource efficiency, and a system-specific measure for platooning.

1) Service Effectiveness:

- **fulfillment Rate (%):** Share of transport requests that are completed within the simulation horizon (i.e., the TU reaches its destination and is unloaded). *Higher is better.*
- **Average Delivery Delay (time units):** Mean delay of completed requests, measured as actual delivery time relative to the baseline plan. Reported over completed requests only; missed jobs are reflected in the fulfillment rate. *Lower is better.*

2) Resource Efficiency:

- **Carrier Utilisation (%):** Fraction of each carrier's available time spent in productive states (loading, unloading, in-transit with TU, or empty repositioning). Idle or waiting time is excluded. *Higher is better.*
- **Empty Travel Ratio (%):** Portion of total carrier travel time spent moving without a TU (repositioning). Indicates deadheading. *Lower is better.*
- **Average TUs per CU:** Throughput per carrier over the horizon—the total number of fulfilled requests divided by the fleet size. Best compared at a fixed fleet size and demand level. *Higher is better.*

3) System-Specific Performance:

- **Platooning Rate (%):** Share of carrier travel time that occurs in coordinated platoons (two or more carriers moving together). Reflects convoying effectiveness; may trade off with delay or empty travel depending on network conditions. *Context-dependent.*

B. Case Study 1: Toy Rail Network

A small-scale, four-station network with bidirectional links was designed to serve as a controlled environment for initial validation (Fig. 5). The network consists of three CUs, and each station has a capacity of two CUs. The primary purpose of this Toy Case was to verify the fundamental logic of the DES-MILP feedback loop and to analyse the system's response to probabilistic disruptions without the confounding variables of a large-scale network. Disruptions were modelled as stochastic events, including CU departure delays (5-10% probability of a 2-9 timestamp delay) and CU breakdowns during loading (20% probability of a 100-150 timestamp outage).

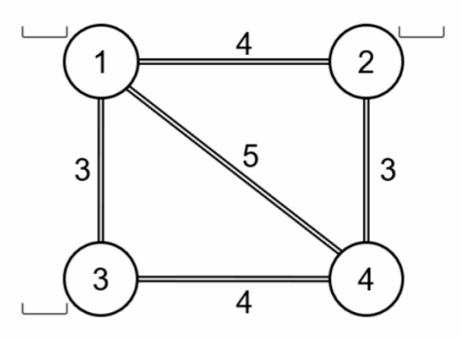

Fig. 5. The four-station Toy Rail Network used for initial validation.

C. Case Study 2: Randstad Rail Network

To assess system resilience under more realistic conditions, a large-scale case study based on the railway network of the Randstad conurbation in the Netherlands was implemented (Fig. 6). This complex network includes 17 stations (9 designated as primary transfer hubs and 8 as routing nodes) and represents a real-world operational environment. The purpose of this case study was to evaluate the framework's ability to handle complex, cascading disruptions. A sequence of deterministic disruptions was applied over a 40-timestamp horizon to test the system's adaptive capabilities. The case study focuses on studies on various disruptions as shown in Table IV

TABLE IV
Disruptions applied in the Randstad Case Study

Disruption Type	Analysis Type
TU Addition	Simultaneous TU addition Non-simultaneous TU addition Time window flexibility Time bracket
CU Breakdown	Single CU breakdown Time window flexibility
Arc Removal	Single arc removal

1) Single Disruption Type Analysis: The analysis begins with individual disruption scenarios to establish baseline performance impacts and system response characteristics.

- **Transport Unit (TU) Addition Disruptions:** The system accommodates dynamic request arrivals at various time points during simulation, including immediate additions at simulation start and mid-simulation arrivals. When

new requests arrive, the system immediately triggers re-optimisation to integrate these demands into the existing plan. The lock-in mechanism preserves all ongoing carrier operations, ensuring that carriers currently executing tasks continue their planned routes while the MILP re-optimises the remaining unassigned requests and newly arrived demands. This approach maintains operational continuity while accommodating dynamic demand changes. The system also supports time bracket analysis, where requests are added at different simulation times to evaluate the impact of timing on system performance, and time window flexibility analysis, where pickup and delivery windows are relaxed to assess the trade-offs between fulfillment and punctuality. Specifically, two distinct TU addition scenarios are analysed: (1) simultaneous addition of three transport units at t=0, and (2) non-simultaneous addition with one transport unit at t=0 and another at t=25, enabling comparison of system performance under different demand surge patterns.

- **Carrier Unit (CU) Breakdown Disruptions:** When carrier units fail during simulation, the system implements a comprehensive lock-in strategy. All active carrier operations are preserved and allowed to complete as planned, while the failed carrier's assigned tasks are immediately dropped from the system. The re-optimisation process then redistributes these dropped tasks among the remaining operational carriers, considering their projected availability times and current workload. This ensures that the system maintains service continuity despite capacity reduction. The system also supports time window flexibility analysis for CU breakdown scenarios, where pickup and delivery windows are relaxed to assess the system's ability to recover from capacity reductions under different flexibility levels. Specifically, the analysis examines a carrier unit breakdown occurring at t=5, evaluating the system's response to mid-simulation capacity reduction.
- **Arc Removal Disruptions:** The system handles both permanent and temporary arc removals. For permanent removals, the arc is unavailable throughout the entire planning horizon, requiring all affected requests to be rerouted through alternative paths. For temporary removals, the arc becomes unavailable during specific time intervals (e.g., from t=7 to t=17), after which it is restored to the network. The lock-in mechanism preserves all ongoing operations that do not depend on the removed arc, while the re-optimisation process reroutes affected requests through alternative network paths, adjusting pickup and delivery windows as necessary to maintain feasibility. This approach preserves operational continuity while adapting to structural network changes.

2) Extended Analysis Scenarios: Building upon the single disruption scenarios, the framework incorporates extended analysis methodologies to provide deeper insights into system behaviour under various operational conditions.

a) Time Window Flexibility Analysis: For both TU addition and CU breakdown scenarios, the system evaluates the impact of time window flexibility on performance metrics. This analysis involves systematically relaxing pickup and delivery time windows by 0, 10, and 40 time units to assess the trade-offs between operational flexibility and service quality. The methodology enables evaluation of how increased flexibility affects fulfillment rates, delivery delays, and overall system efficiency under disruption conditions. Specifically, for TU addition scenarios, two transport units are added at t=14, and then time window flexibility is analysed, while for CU breakdown scenarios, a carrier unit is broken at t=5, and then time window flexibility is analysed to assess the system's recovery capabilities under different flexibility levels.

b) Time Bracket Analysis: The time bracket analysis focuses on TU addition scenarios, examining how the timing of demand surges affects system performance. This methodology involves adding transport units at different simulation time points (t=0, t=10, t=20) to evaluate the system's responsiveness to demand changes occurring at various stages of the planning horizon. The analysis provides insights into optimal timing strategies for accommodating dynamic demand patterns.

3) Multi-Phase Disruption Scenarios: To evaluate system resilience under complex operational conditions, the framework incorporates multi-phase disruption scenarios that combine multiple disruption types in sequence.

a) Sequential Disruption Combination 1: The first combination scenario involves a sequential disruption pattern where CU4 experiences a breakdown at t=11, followed by the addition of TU15 at t=25. This scenario tests the system's ability to handle cascading disruptions, where a capacity reduction is followed by increased demand. The analysis evaluates how the system adapts to the reduced capacity while simultaneously accommodating new demand requirements.

b) Sequential Disruption Combination 2: The second combination scenario represents a more complex multi-phase disruption involving three distinct events: (1) permanent removal of arc (1,3) at simulation start, (2) simultaneous addition of TUs 4, 15, and 18 at t=0, and (3) CU6 breakdown at t=26. This scenario tests the system's resilience under simultaneous structural network changes, demand surges, and capacity reductions. The analysis evaluates the cumulative impact of multiple disruption types and the system's ability to maintain operational continuity under increasingly complex conditions.

V. Results and Analysis

This section presents the key findings from evaluating the DES-MILP framework's resilience to disruptions in the Toy Case (probabilistic disruptions) and the Randstad Case Study (deterministic disruptions), highlighting the system's performance and recovery capabilities under varying conditions.

A. Toy Case: Performance under Probabilistic Disruption

The Toy Case evaluates the DES-MILP framework under probabilistic disruptions across 200 simulation runs, establishing a baseline for system resilience. Disruptions, including

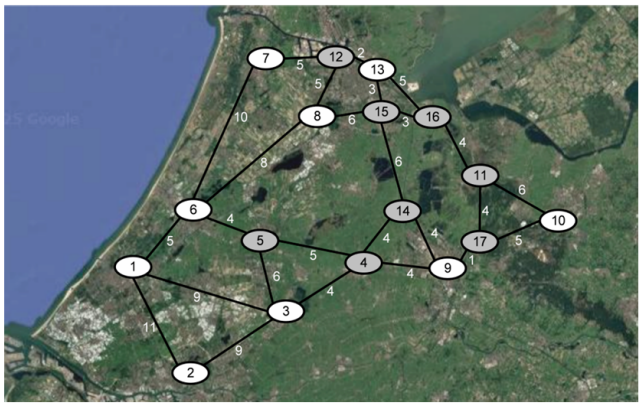

Fig. 6. The Randstad Rail Network case study, based on the existing railway system in the Netherlands.

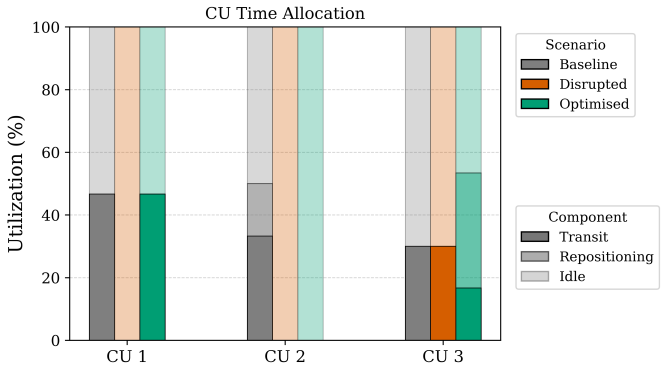

Fig. 7. Carrier unit time allocation for Run 135 under baseline, disrupted, and optimised scenarios.

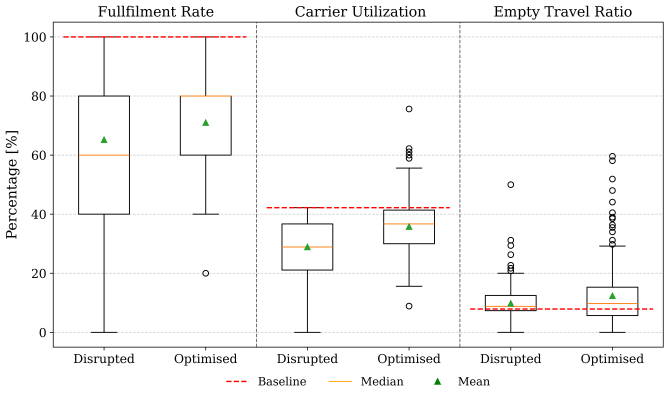

Fig. 8. Distribution of fulfillment rate, carrier utilisation, and empty travel ratio across 200 runs for disrupted and optimised cases. Baseline values are shown as dashed lines.

breakdowns of carrier units (CUs) and transport units (TUs), were applied probabilistically to reflect real-world uncertainties.

For a representative case (Run 135), disruptions occurred with a CU breakdown at time step 2, followed by a TU breakdown at time step 11. These events degraded performance, reducing fulfilled requests from 5 in the baseline to 1 in the disrupted scenario. Re-optimisation improved this to 3 fulfilled requests, demonstrating partial recovery.

TABLE V
KPI SUMMARY FOR THE TOY CASE UNDER PROBABILISTIC DISRUPTIONS (RUN 135).

KPI	Baseline	Disrupted	Re-optimised
Average Delivery Delay	N/A	N/A	3.33
Fulfillment Rate (%)	100.00	25.00	75.00
Carrier Utilisation (%)	42.20	10.00	33.30
Empty Travel Ratio (%)	7.90	0.00	13.40
Platooning Rate (%)	0.00	0.00	3.70

The KPI summary in Table V shows substantial improvements post-optimisation, with fulfillment rate rising from 25% to 75%, carrier utilisation from 10% to 33.3%, and introduction of a 3.7% platooning rate, though at the cost of a higher empty travel ratio (13.4%) due to repositioning. Figure 7 illustrates CU time allocation, revealing complete idleness for CU1 and CU2 in the disrupted case, with CU3 underutilised. Optimisation redistributed workload, restoring CU1 to baseline-like balance and shifting CU3 toward repositioning to enable higher fulfillment.

Figure 8 illustrates the boxplot for 200 simulations. Across all 200 runs, disruptions caused median fulfillment to drop to around 60%, with high variance, while optimisation shifted it to approximately 80%, approaching the 100% baseline. Carrier utilisation followed a similar recovery pattern, though not fully to baseline, and the empty travel ratio increased slightly in optimised cases due to necessary repositioning trade-offs.

Error bars in 9 show mean fulfillment of 71.0% ± 16.8% (optimised) versus 65.2% ± 24.3% (disrupted), indicating reduced variability post-optimisation. Carrier utilization averaged 35.8% ± 11.0% (optimized) and 28.9% ± 10.0% (disrupted), with empty travel at 12.4% ± 10.6% (optimised) and 9.8% ± 7.5% (disrupted).

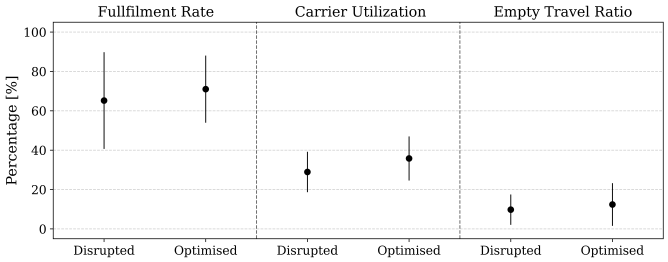

Fig. 9. Errorbar plot of fulfillment rate, carrier utilisation, and empty travel ratio across 200 runs for disrupted and optimised cases.

These results confirm the framework's ability to mitigate stochastic disruptions, restoring significant operational effectiveness through resource reconfiguration, including platooning and repositioning, while highlighting trade-offs in efficiency metrics.

B. Randstad Case Study

1) Single Disruption Type Analysis: This section evaluates the DES-MILP framework's resilience to single disruption types, focusing on key performance indicators (KPIs) such as carrier unit (CU) utilisation, fulfillment rate (%) and empty travel ratio (%) with results derived from percentage differences and visual representations. The analysis begins with a comparative assessment of fulfillment rate, carrier utilisation, and empty travel ratio across four disruption scenarios, followed by a detailed examination of CU utilisation patterns.

TABLE VI
PERCENTAGE DIFFERENCE OF THE RE-OPTIMISED SCENARIO FROM THE BASELINE AND THE DISRUPTED CASE (SINGLE DISRUPTION ANALYSIS)

KPI Name	Baseline	Disrupted
Simultaneous TU Addition		
Fulfillment Rate (%)	-12.00	5.00
Carrier Utilization (%)	-2.30	-2.30
Empty Travel Ratio (%)	-3.00	-3.00
Non-simultaneous TU Addition		
Fulfillment Rate (%)	-5.88	5.88
Carrier Utilization (%)	-3.00	-3.00
Empty Travel Ratio (%)	-1.00	-1.00
CU Breakdown		
Fulfillment Rate (%)	-6.67	6.67
Carrier Utilization (%)	8.50	8.90
Empty Travel Ratio (%)	0.00	-0.80
Arc Removal		
Fulfillment Rate (%)	0.00	13.33
Carrier Utilization (%)	-6.30	0.00
Empty Travel Ratio (%)	-2.00	-1.90

Table VI quantifies the percentage changes in KPIs for the re-optimised scenario relative to baseline and disrupted conditions. For simultaneous TU addition, the fulfillment rate declines by 12% from the baseline but recovers by 5% compared to the disrupted case, accompanied by minor reductions in carrier utilisation (-2.3%) and empty travel ratio (-3.0%). In the non-simultaneous TU addition scenario, the fulfillment rate decreases by 5.88% from the baseline, improves by 5.88% over the disrupted case, with slight decreases in utilisation (-3.0%) and empty travel (-1.0%). The CU breakdown scenario reveals a 6.67% reduction in fulfillment rate from the baseline, a 6.67% improvement over the disrupted case, with notable increases in carrier utilisation (8.5-8.9%) and minimal variation in empty travel ratio (0.0% to -0.8%). Finally, the arc removal scenario maintains baseline fulfillment, achieves a 13.33% improvement over the disrupted case, and exhibits a 6.3% utilisation drop from the baseline alongside minor reductions in empty travel (-1.9% to -2.0%), reflecting effective re-optimisation.

This performance trend is visually supported by Figure 10, which illustrates CU time allocation under the CU breakdown scenario. The figure contrasts baseline (dark grey), re-optimised (green), and breakdown (orange) conditions, highlighting significant idle time (e.g., 100% for CU 4) during the breakdown. Re-optimisation redistributes workload, enhancing active utilisation for CUs such as CU 5 and CU 7,

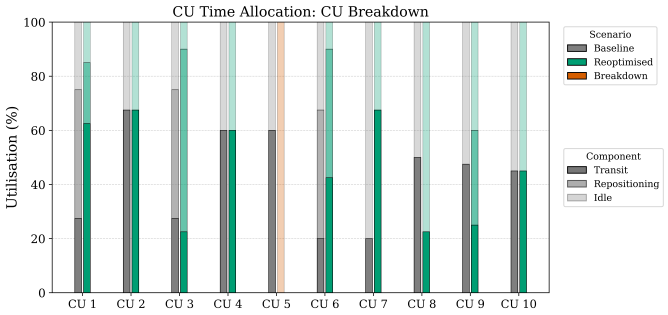

Fig. 10. Carrier Utilisation for CU Breakdown disruption

demonstrating the framework's adaptive capacity. Similarly, Figure 11 presents CU utilisation composition for the arc removal scenario, comparing baseline (grey) and re-optimised (light green) active and idle states. Utilisation shifts range from a -35.0% decrease (CU 1) to a +7.5% increase (CU 5), underscoring the re-optimisation's effectiveness in reallocating resources across the network.

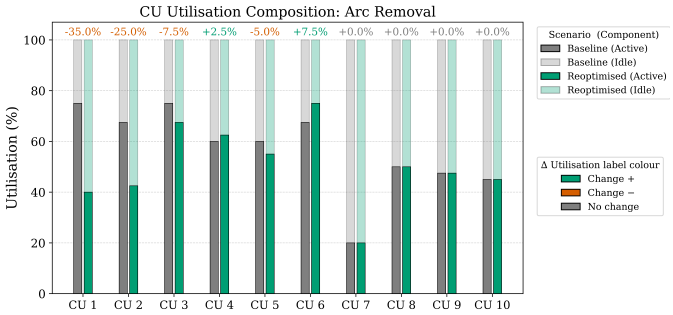

Fig. 11. Carrier Utilisation for CU Breakdown disruption

2) Extended Analysis Scenarios: This section extends the evaluation of the DES-MILP framework by investigating the effects of time brackets and time window flexibility on system performance under specific disruptions, namely TU addition and CU breakdown. The analysis focuses on percentage differences in key performance indicators (KPIs): fulfillment rate, carrier utilisation, and empty travel ratio, for the re-optimised scenario relative to baseline and disrupted conditions, as visualised in the following figures.

Figure 12 illustrates the percentage differences across varying time brackets (t=0, 10, 20, 30) for TU addition disruptions. At t=0 and t=10, the fulfillment rate decreases by 11.10% from the baseline but improves by 5.60% over the disrupted case, with carrier utilisation showing a 6.00% reduction at t=0 and a 12.00% increase at t=10, while empty travel ratio exhibits a 5.70% decline at t=0 and a marginal 0.10% rise at t=10. For later brackets (t=20 and t=30), the fulfillment rate drops by 16.70% from the baseline with no improvement over the disrupted case, carrier utilisation increases by 2.00% at t=20 and remains unchanged at t=30, and empty travel ratio decreases by 1.30% at t=20 with no change at t=30. These trends suggest that earlier time brackets facilitate better recov-

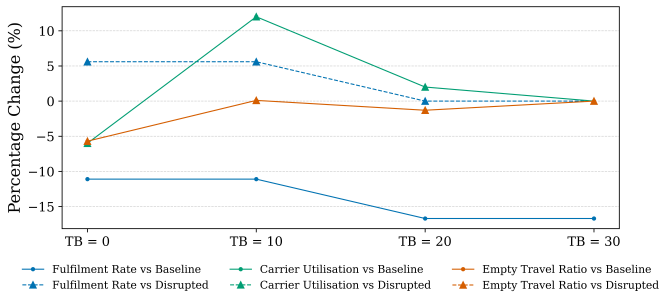

Fig. 12. Percentage Difference of the Re-optimised Scenario from the Baseline and the Disrupted Case (Time bracket analysis for TU Addition)

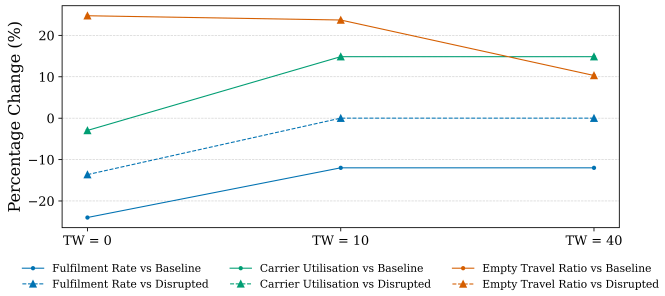

Fig. 13. Percentage Difference of the Re-optimised Scenario from the Baseline and the Disrupted Case (Time window flexibility analysis for TU Addition)

Building on this, Figure 13 examines time window flexibility (t=0, 10, 40) under TU addition disruptions. At t=0, the fulfillment rate declines substantially by 24.00% from the baseline and 13.64% from the disrupted case, with carrier utilisation reducing by 2.97% in both comparisons, contrasted by a 24.74% increase in empty travel ratio. At t=10 and t=40, the fulfillment rate decreases by 12.00% from the baseline with no change relative to the disrupted case, while carrier utilisation rises by 14.85% in both, and empty travel ratio increases by 23.71% at t=10 and 10.31% at t=40. This indicates that greater flexibility in time windows enhances utilisation but may elevate empty travel as a trade-off, with diminishing returns beyond moderate flexibility levels, underscoring the need for balanced parameter tuning in dynamic environments.

Shifting to CU breakdown disruptions, Figure 14 depicts percentage differences across time window flexibility levels (t=0, 10, 40). At t=0, the fulfillment rate reduces by 13.00% from the baseline with no improvement over the disrupted case, carrier utilisation falls by 15.85% and 15.25% respectively, and the empty travel ratio shows no baseline change but a 6.40% decrease from disrupted. At t=10, fulfillment decreases by 7.00% from baseline but improves by 6.90% over disrupted, with utilisation reductions of 8.45% and 7.80%, and empty travel remains unchanged from baseline but down

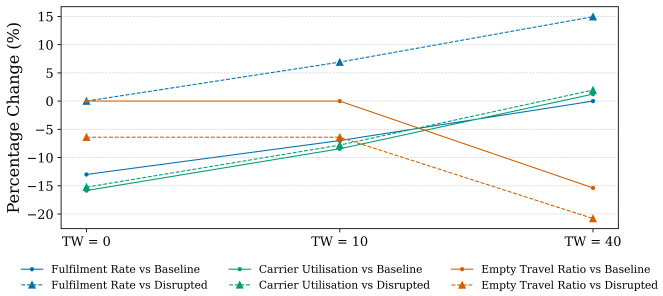

Fig. 14. Percentage Difference of the Re-optimised Scenario from the Baseline and the Disrupted Case (Time window flexibility analysis for CU Breakdown)

6.40% from disrupted. Notably, at t=40, fulfillment matches the baseline and improves by 14.94% over disrupted, utilisation increases by 1.23% and 1.95%, while empty travel declines by 15.38% and 20.80%. These results demonstrate that increased time window flexibility significantly bolsters recovery from CU breakdowns, particularly in fulfillment and utilisation, though at the potential cost of reduced empty travel efficiency, affirming the framework's adaptability in handling resource constraints.

3) Multi-Phase Disruption Scenarios: This section investigates the DES-MILP framework's robustness under multi-phase disruptions, combining CU breakdowns with TU additions and arc removals. The analysis quantifies percentage differences in key performance indicators (KPIs)—fulfillment rate, carrier utilisation, and empty travel ratio—for re-optimised scenarios relative to baseline and disrupted conditions, supplemented by detailed carrier unit (CU) utilisation patterns.

TABLE VII
PERCENTAGE DIFFERENCE OF THE RE-OPTIMISED SCENARIO FROM THE BASELINE AND THE DISRUPTED CASE (MULTIPLE DISRUPTIONS)

KPI Name	Baseline	Disrupted
CU Breakdown + TU Addition		
Fulfillment Rate (%)	-6.25	12.50
Carrier Utilization (%)	-0.80	-0.40
Empty Travel Ratio (%)	-2.60	-3.50
Arc Removal + CU Breakdown + TU Addition		
Fulfillment Rate (%)	-16.67	16.67
Carrier Utilization (%)	-2.40	5.80
Empty Travel Ratio (%)	-5.30	-3.10

Table VII summarises the KPI impacts across two multi-disruption combinations. For CU breakdown combined with TU addition, the fulfillment rate decreases by 6.25% from the baseline but improves by 12.50% over the disrupted case, with minor reductions in carrier utilisation (-0.80% from baseline, -0.40% from disrupted) and empty travel ratio (-2.60% from baseline, -3.50% from disrupted). In the more complex scenario of arc removal plus CU breakdown and TU addition, fulfillment rate drops by 16.67% from the baseline yet recovers by 16.67% relative to disrupted conditions; carrier

utilisation declines by 2.40% from baseline but rises by 5.80% over disrupted, while empty travel ratio decreases by 5.30% from baseline and 3.10% from disrupted. These results indicate that while multi-phase disruptions amplify performance degradation, re-optimisation yields substantial recovery, particularly in fulfillment, though at varying costs to efficiency metrics.

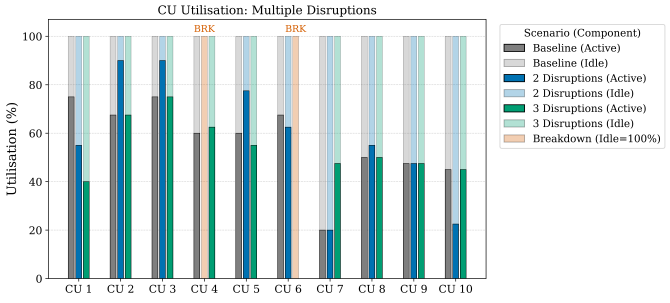

Fig. 15. Carrier Utilisation for multiple disruptions

Complementing this, Figure 15 visualizes CU utilization across individual carriers (CU1 to CU10) under baseline, two-disruption (CU breakdown + TU addition, shown in blue for active and light blue for idle), and three-disruption (arc removal + CU breakdown + TU addition, in green for active and light green for idle) scenarios, with orange bars denoting full idleness (100%) during breakdowns. Compared to baseline (dark grey active, grey idle), the two-disruption case reveals reduced active utilisation for most CUs (e.g., CU1 at approximately 40%, CU4 at 60%) alongside increased idleness, reflecting resource strain from compounded events. The three-disruption scenario exacerbates this, with further drops in active time (e.g., CU5 near 20%, CU9 at 30%) and heightened idleness, yet re-optimisation mitigates some losses by redistributing workloads, as evidenced by slight active utilisation gains in CUs like CU3 and CU7 relative to the disrupted baseline. Overall, the figure underscores the framework's capacity to adapt to escalating disruptions, though cumulative effects progressively challenge utilisation balance, emphasising the need for proactive multi-phase recovery strategies.

VI. Conclusion

This research set out to evaluate and enhance the operational performance of autonomous pod-based railway systems by integrating Mixed-Integer Linear Programming (MILP) with Discrete-Event Simulation (DES) for dynamic evaluation and re-planning. The primary objective was to assess system resilience and the effectiveness of this adaptive strategy under operational disruptions. Analyses on a simplified Toy Case and a large-scale Randstad network across diverse failure scenarios yield the following consolidated insights.

- **Dynamic re-planning is a reliable recovery lever.** On the *Toy Case*, a representative run shows fulfillment rising from **25%** (disrupted) to **75%** (re-optimised), with carrier utilisation increasing from **10%** to **33.3%** and platooning reappearing ($\approx 3.7\%$). Over **200 runs**, re-optimisation shifts the *median* fulfillment from $\sim$**60%** (disrupted) towards $\sim$**80%** (near 100% baseline) and reduces variability—evidence that the loop is consistently corrective rather than case-specific.
- **Resilience entails clear, quantifiable trade-offs.** Service recovery is achieved by reallocating work and tolerating modest efficiency losses. On the *Toy Case*, mean carrier utilisation increases post re-optimisation (**35.8%** $\pm$ 11.0% vs. **28.9%** $\pm$ 10.0%), while the Empty Travel Ratio ticks up (**12.4%** $\pm$ 10.6% vs. **9.8%** $\pm$ 7.5%), reflecting deliberate deadheading to restore service. In *Randstad*, similar patterns appear: utilisation generally climbs relative to the disrupted state, while empty travel stays flat or slightly rises as the fleet reconfigures.
- **Single disruptions are recoverable; arc removals are structurally limiting.** In *Randstad* single-shock scenarios (e.g., TU additions, CU breakdown), re-optimisation consistently lifts fulfillment by $\sim$**+5–13.33%** over the disrupted state and rebalances CU workloads. *Arc removal* is the hardest to offset: fulfillment improves vs. disrupted ($\sim$**+13.33%**) but cannot fully return to baseline utilisation ($\sim$**–6.3%**), revealing hard network-topology limits that optimisation alone cannot erase.
- **Time-window flexibility is a powerful enabler.** Allowing moderate slack in delivery windows after a CU breakdown returns fulfillment to (or above) baseline (e.g., $\sim$**+14.94%** vs. disrupted), nudges utilisation upward, and cuts empty travel markedly (about **–15%** vs. baseline and **–21%** vs. disrupted). Small contractual flexibilities therefore convert directly into operational robustness.
- **Compound disruptions remain manageable—up to a point.** Under sequential disruptions, the framework still yields clear gains over the disrupted state even if baseline levels are not always reachable (e.g., $\sim$**+12.50%** fulfillment recovery for CU breakdown + TU addition; $\sim$**+16.67%** in a triple disruption with arc removal). Workload redistribution is visible at the asset level: failed CUs go idle while healthy units absorb uneven but stabilising loads.

A. Industrial, Policy, and Managerial Insights

a) Policy directions: Policymakers should allow a small amount of delivery-window flexibility. Even a little slack helps raise post-disruption fulfillment, though it may reduce punctuality. Price this clearly with two service levels (tight vs. flexible windows) and set a maximum allowed lateness for each level. Encourage early TU bookings with booking cut-offs and small discounts or credits. Treat digital re-planning as basic resilience infrastructure: set minimum standards such as a maximum time to produce a new plan, an audit trail of decisions, and certified training for control-room staff. Because losing an arc puts a hard limit on recovery, invest in backup routes—loops and short detours—on fragile, high-volume corridors, supported by scenario-based cost–benefit analysis [12]. Finally, require a standard public report so

results are comparable: fulfillment, Average Delay, Carrier Utilisation, Empty Travel Ratio, and Platooning Share from event-coded logs.

b) Managerial Implications and Operational Guidance: Managers should write a simple re-optimisation playbook and use it by default. Re-optimise when a breakdown, late TU, or arc closure occurs; also re-optimise at a fixed rhythm (for example, every 5–10 minutes). Define clear escalation rules when shocks stack up. Keep a small standby buffer (think $N-1$) and stage carriers at merge or transfer nodes to keep platoons and cut empty moves. Offer two products: *Premium* (tight times) and *Flexible* (wider times with better fulfillment under disruption). Steer demand with early-booking incentives and, if needed, surcharges or cut-offs for late requests. Watch workload balance at CU level; if a few units do most of the work, rotate or rebalance. Prepare detour playbooks for critical OD pairs and practise them with the infrastructure manager. Run a live KPI dashboard (fulfillment, Delay, Utilisation, Empty Ratio, Platooning) with thresholds that automatically trigger a re-optimisation cycle when limits are crossed [14].

B. Contributions, Limitations and Future Research

a) Contributions: This work delivers a practical, event-driven re-optimisation loop for autonomous pod rail by tightly coupling DES (execution) with MILP (planning). Key contributions are:

- **Integrated DES–MILP recovery loop:** A rolling re-optimisation with clear lock-in rules that bridges static plans and dynamic execution, enabling fast recovery when disruptions occur.
- **Formal state–event model and KPI protocol:** TU/CU state–event definitions with event-coded logs that make fulfillment, delay, utilisation, empty travel, and platooning reproducible and comparable across runs.
- **Disruption design and evidence:** A structured set of shocks (TU additions, CU breakdowns, arc removals; single and sequential) showing consistent service recovery and revealing service–efficiency trade-offs.
- **Policy/managerial levers validated:** Clear effects of bounded time-window flexibility, plus operational doctrines (lock-ins + rolling re-optimisation) that improve resilience.
- **Demonstration on two scales:** A Toy Case (200-run statistics) and a large Randstad network, including platooning representation and empty-carrier relocation behaviour.

b) Limitations and Future Research: While the framework is robust, several limits suggest next steps:

- **Data realism:** Results are based on synthetic data; validate with real demand traces, operational logs, and observed disruption patterns.
- **Stochasticity and capacity:** Extend from deterministic travel/handling times to stochastic models; add station/track capacity and queueing effects.
- **Horizon and scale:** The current horizon (40 time steps) limits long-run effects; benchmark run-times and scalability on larger networks and longer horizons, exploring decomposition or parallelisation.
- **Objectives and metrics:** Add cost, energy, and emissions to the KPIs; study multi-objective trade-offs and pricing of flexibility in SLAs.
- **Proactive strategies:** Evaluate pre-positioning, standby buffers, predictive triggers, and learning-based (e.g., RL) re-planning policies alongside the MILP–DES loop.
- **Design of lock-ins and triggers:** Test alternative lock-in rules and re-optimisation trigger policies to quantify robustness across operating policies.
- **Multi-actor coordination:** Model operator–infrastructure interactions for detour design and redundancy planning; study incentive schemes for early TU commitments.

In conclusion, coupling optimisation with simulation provides a practical, interpretable pathway to resilient autonomous freight operations. The framework not only evaluates performance under stress but actively improves it by triggering immediate re-optimisation, exploiting limited schedule slack, and revealing when structural constraints (e.g., arc removals) require network redundancy beyond algorithmic recovery. These findings suggest three operating guidelines: **(i)** trigger automated re-optimisation at disruption onset, **(ii)** systematize limited time-window flexibility in SLAs, and **(iii)** invest in routing redundancy where topology is fragile.

REFERENCES

[1] Pods4Rail. (2024) Homepage 01 - pods4rail. [Online]. Available: https://pods4rail.eu/

[2] A. Di Febbraro, N. Sacco, and M. Saeednia, "One-way car-sharing profit maximization by means of user-based vehicle relocation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 2, pp. 628–641, 2018.

[3] A. D. Febbraro, N. Sacco, and M. Saeednia, "One-way car-sharing profit maximization by means of user-based vehicle relocation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 2, pp. 628–641, 2018.

[4] X. Liao, J. Han, S. Mahnam, and A. Paz Martinez, "Unlocking the potentials of modularity in railways, a heuristic framework for pods scheduling," in *27th IEEE International Conference on Intelligent Transportation Systems, ITSC 2024*. IEEE - Institute of Electrical and Electronics Engineers, 2024.

[5] A. D. Febbraro, N. Sacco, and M. Saeednia, "One-way carsharing," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2319, no. 1, pp. 113–120, 2012.

[6] Pods4Rail, "Homepage 01 - Pods4Rail," 12 2024. [Online]. Available: https://pods4rail.eu/

[7] X. Liao, J. Han, and M. Saeednia, "Modular vehicle routing on railways: Opportunities for intermodality," 2024.

[8] A. Tafreshian, N. Masoud, and Y. Yin, "Frontiers in service science: Ride matching for peer-to-peer ride sharing: A review and future directions," *Service Science*, vol. 12, no. 2, pp. 44–60, 2020.

[9] Y. H. Wu, L. J. Guan, and S. Winter, "Peer-to-peer shared ride systems," in *Lecture Notes in Computer Science*, 2008, pp. 252–270.

[10] R. Ma, L. Yao, L. Song, and M. Jin, "A novel algorithm for peer-to-peer ridesharing match problem," *Neural Computing and Applications*, vol. 31, pp. 247–258, 2019.

[11] X. Bei and S. Zhang, "Algorithms for trip-vehicle assignment in ride-sharing," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[12] A. Alfieri, R. Groot, L. Kroon, and A. Schrijver, "Efficient circulation of railway rolling stock," *Transportation Science*, vol. 40, no. 3, pp. 378–391, 2006.

[13] M. Peeters and L. Kroon, "Circulation of railway rolling stock: a branch-and-price approach," *Computers & Operations Research*, vol. 35, no. 2, pp. 538–556, 2 2007.

[14] S. Illgen and M. Höck, "Literature review of the vehicle relocation problem in one-way car sharing networks," *Transportation Research Part B: Methodological*, vol. 120, pp. 193–204, 2019.

[15] M. Clemente, M. P. Fanti, A. M. Mangini, and W. Ukovich, "The vehicle relocation problem in car sharing systems: Modeling and simulation in a petri net framework," in *Application and Theory of Petri Nets and Concurrency*. Springer Berlin Heidelberg, 2013, vol. 7927, pp. 250–269.

[16] S. Weikl and K. Bogenberger, "Relocation strategies and algorithms for free-floating car sharing systems," *IEEE Intelligent Transportation Systems Magazine*, vol. 5, no. 4, pp. 100–111, 2013.

[17] E. M. Cepolina, A. Farina, and A. Pratelli, "Car-sharing relocation strategies: A state of the art," in *WIT Transactions on State-of-the-Art in Science and Engineering*, 2014, pp. 109–120.

[18] A. Ait-Ouahmed, D. Josselin, and F. Zhou, "Relocation optimization of electric cars in one-way car-sharing systems: modeling, exact solving and heuristics algorithms," *International Journal of Geographical Information Science*, vol. 32, no. 2, pp. 367–398, 2018.

[19] P. J. Eder, S. Ramoser, S. Braun, and S. Weltge, "Efficient active–passive vehicle coordination in multimodal transportation networks," *Transportation Research Part E: Logistics and Transportation Review*, vol. 203, p. 104318, 2025.

[20] N. D. Versluis and M. Saeednia, "Modelling rail carrier assignment and relocation in multimodal pod system," in *Proceedings of the 5th International Rail Symposium Aachen (IRSA 2025)*, 2025.

Algorithm 1 Overview of DES logic

Require: *tu_data*: list of transport-unit records
Require: *cu_data*: list of carrier-unit records
Require: *platoon_arcs*: set of $(s_1,\ s_2,\ t_0)$ where platooning occurs
Ensure: *trace* $\leftarrow$ []

procedure MAIN
  for all *tu* $\in$ *tu_data* **do**
    SPAWN TU_PROCESS(*tu*)
  end for
  for all *cu* $\in$ *cu_data* **do**
    SPAWN CU_PROCESS(*cu*)
  end for
  SIMULATEUNTILALLDONE
  return *trace*
end procedure

function TU_PROCESS(*tu*)
  wait until *tu.pickup_time*
  log E1: "TU arrives at station"
  wait *tu.load_time*
  log E4: "load"
  log E5: "depart loaded"
  wait *tu.delivery_time - tu.pickup_time*
  log E8: "arrive delivery"
  wait *tu.unload_time*
  log E9: "unload"
end function

function CU_PROCESS(*cu*)
  Initialise CU state
  log E2: "CU becomes available"
  log E3: "CU arrives at start"
  for all *arc* $\in$ *cu.itinerary* **do**
    $(s_1, s_2, t_0, dur) \leftarrow$ (from, to, start_time, travel_time)
    assignedTU $\leftarrow$ (is there a TU on $(s_1 \rightarrow s_2,\ t_0)$?)
    inPlatoon $\leftarrow (s_1,\ s_2,\ t_0) \in$ *platoon_arcs*
    wait until t_0
    if assignedTU **then**
      log E4: "load TU"
      log E5: "depart loaded" (solo or platoon if inPlatoon)
    else if not assignedTU **then**
      log E10: "depart empty" (solo or platoon if inPlatoon)
    end if
    wait dur
    if assignedTU and this is the TU's drop-off arc **then**
      log E8: "arrive delivery"
      log E9: "unload"
    else if inPlatoon and next arc not in platoon_arcs **then**
      log E7: "leave platoon" (solo or at station)
    else
      log E3: "arrive at station" (repositioning)
    end if
  end for
end function

Algorithm 2 Sequential Disruption Framework for DES-MILP Integration

Require: Disruption configuration $\mathcal{D} = \{(d_1, t_1), (d_2, t_2), \ldots, (d_n, t_n)\}$, Total horizon T

Extract baseline MILP inputs from case study
Set $t_{\text{opt}} = 0$
for all i in $\mathcal{D}$ **do**
    $t_i \leftarrow$ disruption time
    if $t_i = t_{\text{opt}}$ **then**
        Apply disruption logic to inputs at $t = t_{\text{opt}}$
        Prepare disrupted MILP inputs
        Run MILP optimisation to generate plan file (.pkl)
        $t_{i+1} \leftarrow$ next disruption time
        if t_{i+1} exists **then**
            Simulate DES from t_{opt} to t_{i+1} with lock-in method
            Receive Excel output from t_{opt} to t_{i+1}
            Extract MILP inputs at disruption time t_{i+1}
            $t_{\text{opt}} \leftarrow t_{i+1}$
            continue
        else
            Simulate DES from t_{opt} to T without lock-in
            Generate Excel output from t_{opt} to T
        end if
    else
        Run MILP optimisation to generate plan file (.pkl)
        Simulate DES from t_{opt} to t_i with lock-in method
        Receive Excel output from t_{opt} to t_i
        Extract MILP inputs at disruption time t_i
        Apply disruptions at $t = t_i$
        Run disrupted MILP from t_i to T to generate plan file (.pkl)
        $t_{i+1} \leftarrow$ next disruption time
        if t_{i+1} exists **then**
            Simulate DES from t_i to t_{i+1} with lock-in method
            Receive Excel output from t_i to t_{i+1}
            Extract MILP inputs at disruption time t_{i+1}
            $t_{\text{opt}} \leftarrow t_{i+1}$
            continue
        else
            Simulate DES from t_i to T without lock-in
            Generate Excel output from t_i to T
        end if
    end if
end for
Aggregate performance metrics across all simulation segments
Calculate system resilience indicators