ReMuNet

Pioneering resilient and adaptive multimodal transport networks

D4.2

Goal system, objective functions and KPIs

Fraunhofer Austria Research GmbH

19/12/2025

Funded by the European Union

PROJECT INFORMATION

PROGRAMME	Horizon Europe
TOPIC	HORIZON-CL5-2022-D6-02-07
TYPE OF ACTION	HORIZON Research and Innovation Actions
PROJECT NUMBER	101104072
START DAY	1 July 2023
DURATION	36 months

DOCUMENT INFORMATION

TITLE	Goal system, objective functions and KPIs
WORK PACKAGE	4
TASK	T 4.2 Define optimization goals of multimodal transport networks
AUTHORS (Organisation)	Catherine Laflamme (FhA), Thomas Kroiss (FhA), Felix Kamhuber (FhA), Sandra Stein (FhA)
REVIEWERS	All Partners
DATE	1. December 2025

DISSEMINATION LEVEL

PU	Public, fully open	x
SEN	Sensitive, limited under the conditions of the Grant Agreement	
Classified R-UE/EU-R	EU RESTRICTED under the Commission Decision No2015/444	
Classified C-UE/EU-C	EU CONFIDENTIAL under the Commission Decision No2015/444	
Classified S-UE/EU-S	EU SECRET under the Commission Decision No2015/444	

DOCUMENT HISTORY

VERSION	DATE	CHANGES	RESPONSIBLE PARTNER
1.0	01/10/2025	Preliminary version for review by partners	Catherine Laflamme (FhA), Thomas Kroiss (FhA), Sandra Stein (FhA)
1.1	05/12/2025	Extension of chapters 1, 3 and 5	Felix Kamhuber (FhA), Thomas Kroiss (FhA)
1.2	12/12/2025	Review of chapters 1-5	Felix Kamhuber (FhA), Thomas Kroiss (FhA)
1.3	17/12/2025	Final version and internal submission	Felix Kamhuber (FhA)
1.4	19/12/2025	Final version and external submission	Sandra Stein (FhA)

LEGAL NOTICE

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or CINEA. Neither the European Union nor the granting authority can be held responsible for them.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

RL	Reinforcement Learning
STP	Synchromodal Transport Re-Planning Problem
ALNS	Adaptive Large Neighborhood Search
GA	Genetic Algorithm
MARL	Multi-Agent Reinforcement Learning
MDP	Markov Decision Process
HH	Hyper Heuristics
HRL	Hybrid Reinforcement Learning

Executive Summary

1. Goal System (Problem Setup)

The planned problem studied in the research project ReMuNet is described as **S**ynchromodal **T**ransport Re-**P**lanning Problem (STP). The given problem is stochastic, dynamic and allows for online transport re-planning.

The planning scenario describes a transport network with a fixed vehicle timetable and an existing feasible transport plan as an initial solution, which assigns orders to vehicles and routes. The stated problem includes possible disruptions (e.g. disturbances, delays) during transport. When a disruption occurs, a re-planning process is triggered that involves two main steps:

1. Decision on necessity: Determine whether a specific order needs to be re-planned due to the disruption (e.g., missed arrival date)
2. Decision on alternatives: If re-planning is required, select the best alternative vehicle and route combination to serve that order within the updated operational constraints

A comprehensive overview of Operational Synchromodal Transport Planning publications is provided in [1]. These can be divided into heuristic (34%), solver-based (28%), Approximate and Exact (each 16%) and a small proportion of Learning-Based (7%) solutions. Learning-based approaches are sub-divided into Departure Learning (25%) and Reinforce Learning (RL) approaches (75%). While exact solutions solve the problem precisely, heuristic solutions focus only on near-optimal solutions. The “learning-based” approaches use machine learning techniques to recognize patterns, improve decision-making processes, or enhance the efficiency and accuracy of solving OSTP problems by leveraging data-driven insights and predictive models.

Resilient planning, on of the core challenges, emphasizes the system's ability to maintain its functionality and quickly return to normal operation after a disruption (see Figure 1 from [1]), whereby most STP studies perform replanning after the Time of Disruption (TD).

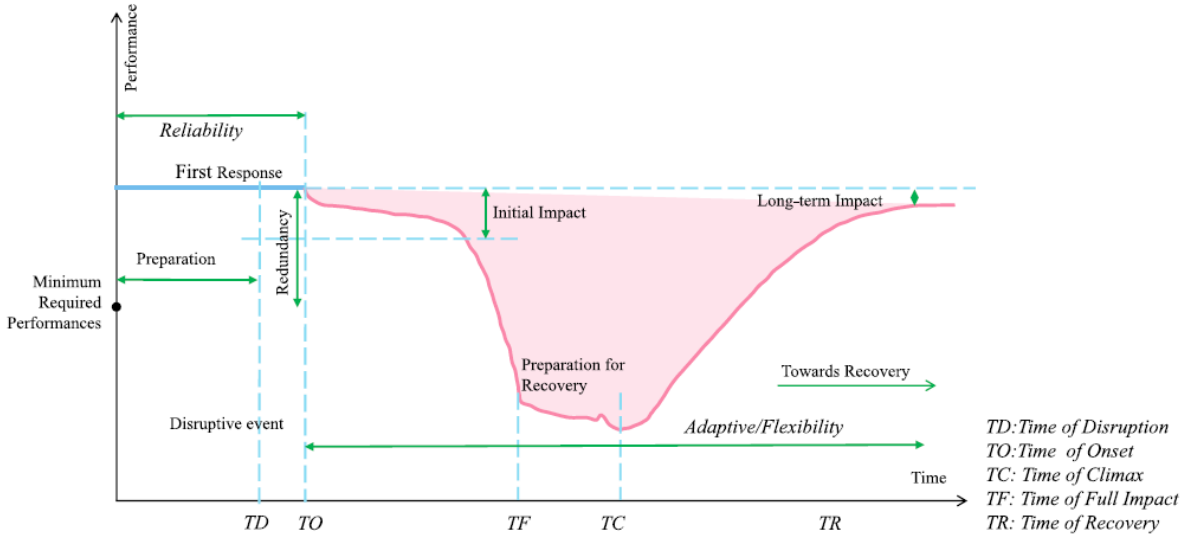

Figure 1: Performance of a resilient synchromodal transport system against disruptions at different stages along the timeline

The problem investigated excludes prior knowledge of the specific demand in the given network. The physical transport network covering three transport modes (water, rail and road) is described as directed multigraph $N = (H, P)$. H is the set of terminals and P is the set of modal transport routes (paths or edges within the graph). Each edge or path $p \in P$ in the graph represents a directed physical transport route that can be travelled either by road, rail, waterway, or any combination thereof and connects two terminals (or nodes) jh_i and h_j.

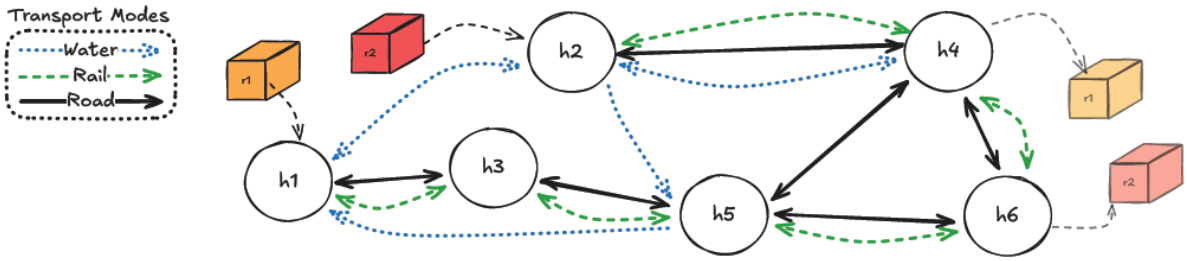

Figure 2: A simplified visualization of an exemplary physical network

Terminals are illustrated as circles in Figure 2 – the arrows represent the edge shape in form of modal transport connections. Furthermore, two exemplary transport requests with their pickup and delivery terminals are shown. A defined number of vehicles $v \in V$ operates on the set of paths in the transport network on a defined schedule.

1.1 Input Data

The graph is represented by its Excel data input including individual sheets for:

- **Carrier data:** Carriers with their attributes travelling on the edges of the STP. Each carrier is described by its mode (rail, road, water), capacity, CO_2 emissions (measured in the amount of CO_2 g per km), maximum velocity
- **Hubs:** Storage capacity, handling rate, handling slots, latitude and longitude
- **Paths:** Start and end hub name, mode, distance (travelled) and maximum velocity along its path. The carrier velocity is restricted by the path velocity.
- **Disruptions:** Amount if initially set disruptions. Additional disruptions occur during the progress of the simulation
- **Timetable:** Each timetable entry id includes two lines for time of departure on the start hub and time of arrival on the end hub for a specific carrier. While the schedule is initially set statically, delays may occur during a trip, causing the schedule for a specific connection on a specific vehicle to change.
- **Orders:** Each order has its own ID, start and end hub, its earliest start time and due time and the amount of transport units.
- **Order Assignments:** Each order has its initial schedule which may be changed during the replanning.

1.2 Costs

The main objective of the given STP problem is to assign demand requests to physical paths and vehicles to derive long-term cost-optimal scenarios. The cost function consists of several components such as costs for transit and transfer, storage (in a hub), waiting times, carbon tax (regarding CO_2 emissions) and penalties for delays [1].

Transport and emission costs can be calculated immediately for each action by a chosen vehicle for a defined route (path) between two terminals. The cost function for penalties can only be calculated when an order has arrived at its final destination terminal. If an order is already late (e.g., when arriving at an intermediate hub), a cost approximation for penalties is calculated. In practice the cost of delay should outweigh the costs arising from transport and emissions to ensure that orders are not trivially left idling in hubs.

The (intermediate) cost evaluation for the current solution includes the following steps:

1. **Dynamic Transport Costs:** For each edge in the transport plan, the cost is calculated as the product of the edge's distance and its cost per unit distance (*edge.distance * edge.cost*).
2. **Fixed Transport Costs:** Fixed costs are added based on the transport mode (e.g., road, rail or water). These are retrieved from the corresponding dictionary.
3. **The total cost** for each order is the sum of the dynamic transport cost and the fixed transport cost for all edges in the order's transport plan.

In case of detected scheduling conflicts (e.g., overlapping edges) a large penalty value is added to the total cost to indicate infeasibility. This concept ensures that the cost evaluation accounts for both feasible and infeasible transport plans. For further information and implementation details (regarding KPI) see chapter 5.

1.3 Time

The RL agent's core objective is to ensure that orders arrive on time at their destination. The simulation is interrupted at each intermediate arrival of an order (with a freight) on a hub at a specific time. The delivery time for each order at each hub is measured using the track_*order* attribute of the order object. The following steps are necessary during the progress of the simulation:

1. Each time an order arrives at a hub, the arrival time and the hub ID are recorded in the *track_order* list.
2. If the order departs from a hub, the departure time and the hub ID are also recorded in the *track_order* list.
3. The last entry in the *track_order* list is used to determine if the order has reached its final hub.
4. The *synchronize_next_hub* method is used to calculate the expected arrival or departure time at the next hub based on the timetable data.

This mechanism ensures that both actual and expected times are tracked and updated for each order at each hub. It basically serves as measurement criteria for (intermediate) order tardiness.

1.4 Emissions

Synchromodal transport is a real-time, dynamic, and optimized intermodal system aimed at improving efficiency and reducing emissions by shifting freight to sustainable modes like rail and waterways while enabling real-time re-planning and routing [8].

In our approach the RL agent also minimizes emissions by favoring train over truck transports, see section 4.2. CO_2 emissions in our approach are evaluated in the *evaluateSimKPIs* method of the *SimKpis* class. The evaluation steps include:

1. **Iterate Through Order Assignments:** For each order assignment, the method retrieves the corresponding edge data from the simulation graph

2. **Retrieve CO_2 Emissions:** The CO_2 attribute of the edge data is accessed, which represents the CO_2 emissions for that specific transport segment.
3. **Aggregate CO_2 Emissions:** The CO_2 emissions for all edges in an order's transport plan are summed up to calculate the total CO_2 emissions for the order.
4. **Handle Infeasible Plans:** If a scheduling conflict is detected (e.g., overlapping edges), a large penalty value (large_co2) is added to the total CO_2 emissions to indicate infeasibility.

This ensures that the CO_2 emissions are calculated accurately for feasible plans and penalized for infeasible ones.

1.5 Resilience

The core objective is to optimize costs, time, and emissions in the long-term. Another target of our use-case implementation is to increase resilience in synchromodal transport. This is achieved by reacting swiftly to the disruptions posed while maintaining the initial transport route as far as possible, see figure 1. Resilience can be measured and evaluated implicitly and relatively in various scenarios with different levels of disruption (minimal, moderate, severe) by comparing each scenario with the initial solution or with each other. The criteria used are the number of planned arrivals (on order level) per scenario and the number of orders that missed their due date. The ability to replan under uncertainty is one of the key reasons for applying RL in the project context. Another characteristic of resilience is that the agent does not have to reschedule at the slightest disruption, e.g., that sufficient buffer time is reserved when switching from one edge to the next at a hub.

2. General Introduction to RL

Reinforcement Learning (RL) is a core discipline within artificial intelligence and machine learning that addresses the challenge of learning how to act to achieve long-term objectives. At its foundation, RL describes a process where an **agent** interacts with an **environment** through a continuous cycle of decisions and feedback. The agent selects an action, the environment responds by moving into a new state, and a **reward signal** is provided that quantifies the desirability of the outcome. Over many iterations, the agent learns strategies - **policies** - that improve its decision-making with the goal of maximizing cumulative reward.

What distinguishes RL from other machine learning paradigms is its reliance on **trial-and-error learning** rather than on explicit supervision. In supervised learning, the “correct” label or output is provided for each input, but in RL the agent must discover effective strategies without being told what the optimal action is in advance. The only feedback is whether a chosen action leads to positive or negative results, and this delayed and indirect feedback is what makes RL both powerful and challenging.

The general RL cycle can be summarized as follows:

1. The agent observes the current **state** of the environment.
2. Based on its current knowledge or policy, the agent selects an **action**.
3. The environment transitions to a new **state** and emits a **reward**.
4. The agent updates its knowledge to improve future decisions.

This process repeats continuously, enabling the agent to gradually refine its policy. Because of this iterative nature, RL is particularly effective in complex and dynamic environments where conditions change and where the consequences of decisions may only become apparent after multiple steps.

Applications of RL span a wide range of fields, including robotics, healthcare, recommendation systems, financial trading, and logistics. Its strength lies in its adaptability: the same principles that allow an RL agent to learn how to play games at superhuman levels can also be applied to optimize transport networks, supply chains, and route planning. For logistics, RL is valuable because it can weigh multiple competing objectives—such as time, cost, energy consumption, and reliability—while continuously improving strategies through interaction with the system.

2.1 Theory behind RL

The theoretical foundation of Reinforcement Learning is often formalized using the mathematical framework of a **Markov Decision Process (MDP)**. A MDP provides a structured way to represent decision-making problems where outcomes are partly random and partly under the control of the agent. The main elements are:

- **States (S):** A set of possible situations in which the environment can be.
- **Actions (A):** The set of all possible decisions the agent can take.
- **Transition function (P):** The probability distribution that determines how the environment moves from one state to another, given a chosen action.
- **Rewards (R):** A function that assigns a numerical value to each state-action transition, representing the immediate benefit or cost.
- **Policy (π):** A strategy that defines the agent's behavior by mapping states to actions.
- **Discount factor (γ)**, which balances short-term and long-term rewards.

A MDP implicitly assumes the Markov property: the next state distribution depends only on the current state and chosen action. Many real-world routing problems are only approximately Markovian (partial observability), in which case a Partially Observable MDP (POMDP) or augmentation of the state with history may be used.

Some major classes of methods in RL theory are stated as the following:

1. **Value-based methods**: These focus on estimating the expected return of being in a particular state or taking particular action (e.g., Q-learning). The agent acts by choosing actions that maximize these value estimates.
2. **Policy-based methods**: These directly optimize the policy itself, often using gradient-based optimization techniques. Such methods can handle complex or continuous action spaces more effectively.
3. **Actor-Critic methods**: These combine both value-based and policy-based approaches, where one component (the critic) evaluates actions while another (the actor) updates the policy.

A critical theoretical challenge in RL is the **exploration-exploitation trade-off**. The agent must balance the need to **explore** new actions to gain information about the environment with the need to **exploit** its current knowledge to maximize rewards. If it only exploits, it risks missing better strategies; if it only explores, it wastes opportunities for improvement.

Another key concept is **convergence**: under certain conditions, RL algorithms are proven to converge to an optimal policy. However, in real-world problems—especially in large or complex environments—exact convergence may be impractical, and approximate methods are employed. This leads into modern advancements such as **Deep Reinforcement Learning (Deep RL)**, where neural networks are used to approximate value functions or policies in high-dimensional spaces.

In summary, the theoretical framework of RL provides a robust foundation for solving sequential decision-making problems. By grounding itself in probability, optimization, and control theory, RL offers both the mathematical rigor to model complex environments and the flexibility to adapt to practical, real-world applications such as logistics optimization.

2.2 SOTA RL for Routing

Reinforcement learning (RL) can be used for transport routing by framing routing decisions as a sequential decision-making problem. In this setting, the environment is the transport system (roads, vehicles, terminals, demand, disruptions), the agent is a vehicle or fleet manager, and the actions are routing choices such as selecting the next stop, allocating a shipment to a mode, or re-routing around disruptions. The agent learns a policy - a mapping from states of the system (traffic conditions, vehicle positions, demand levels) to routing actions – through interaction with the environment.

Each action leads to a new state and produces a reward signal that reflects operational goals, such as minimizing travel time, costs, delays, or emissions, or maximizing service reliability. Over time, the agent improves its policy by maximizing long-term cumulative rewards. In practice, RL allows routing strategies to adapt dynamically to real-time changes in demand, congestion, or disruptions, rather than relying only on static or pre-computed plans.

In vehicle routing, RL can therefore be used to decide the order in which customers are served, how to re-route vehicles when conditions change, or how to coordinate fleets across multimodal networks. Compared to traditional optimization methods, the main appeal of RL is its flexibility: once trained, an RL agent can make fast decisions online and adjust to unforeseen events without solving a new optimization problem from scratch.

Recent years have seen growing interest in applying reinforcement learning (RL) to routing and planning problems within synchromodal and multimodal transport systems. RL's ability to learn adaptive policies in dynamic, high-dimensional environments make it a promising candidate for real-time re-planning and robust decision-making. Early RL-based approaches include Deep Q-Network (DQN) implementations for assignment problems in multimodal networks, Q-learning for synchromodal matching, and hybrid methods that combine Deep RL with local search heuristics for the Vehicle Routing Problem (VRP). Model-assisted RL strategies, such as integrating Adaptive Large Neighborhood Search (ALNS) with DRL, have also been proposed to address uncertainties like service times at terminals. While these RL-based methods demonstrate potential for handling disruptions and improving resilience, they often face challenges in scalability and limited action spaces, which can restrict their effectiveness in large or complex networks. Compared to traditional planning methods (e.g., dynamic control, column generation, genetic algorithms, and hybrid simulation-optimization) RL approaches offer greater flexibility but still require further development to fully exploit their advantages in operational settings.

3. Applying RL for our Goal System

Figure 3 (own representation) compares four different RL approaches (Global RL, Multi-Agent RL, Hierarchical RL and Hybrid RL) for synchromodal transport problems. In the implementation of our hybrid RL approach, a selection of pre-defined rules (heuristics) is used to control the agent through our RL transportation network.

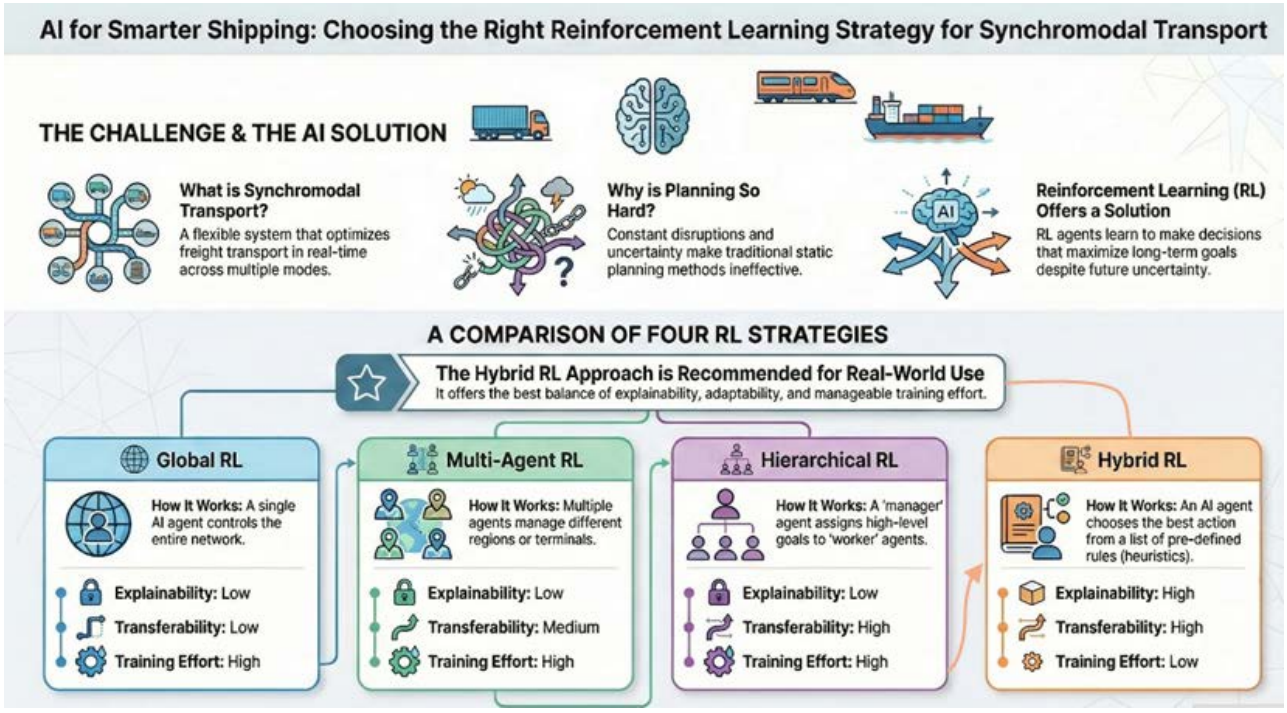

Figure 3: Comparison of RL strategies for STP problems

The following section will discuss the advantages and disadvantages of each approach in a comparative manner.

3.1 Global RL

The Global Reinforcement Learning (RL) approach involves deploying a single RL agent with access to the complete state information of the transport network. This agent is responsible for making all routing and vehicle-assignment decisions across the network. Upon the arrival of an order at any terminal, the agent determines the most suitable vehicle for transportation and plans the route to the destination terminal. By centralizing decision-making, the agent can theoretically optimize the network holistically, potentially discovering novel strategies that improve overall performance. This concept draws inspiration from the success of RL in complex board games such as Go and Chess, where agents have achieved superhuman performance by leveraging full knowledge of the game state.

However, the application of Global RL in transport networks presents significant computational challenges. Training such an agent requires substantial resources, including deep neural network architectures and extensive training time on specialized hardware. Moreover, unlike static board games, transport networks are highly dynamic, with variable elements such as vehicles, orders, timetables, and disruptions. This variability increases the complexity of the learning task and further amplifies computational demands. While the Global RL approach has the potential to yield optimal solutions, its practicality in real-world scenarios is limited by scalability, adaptability, and explainability concerns. In operational contexts, it may be preferable to accept a trade-off between optimality and efficiency, favoring approaches that offer faster learning, greater generalizability, and improved transparency.

3.2 A Hierarchical Approach

HRL divides decision-making into multiple levels of abstraction, unlike MARL, where agents act at the same level. In HRL, high-level agents (managers) assign subtasks to low-level agents (workers), who execute concrete actions to fulfill these tasks. Each level solves its own Markov Decision Process (MDP).

This structure works well for problems with long planning horizons or sparse rewards, as high-level decisions simplify exploration and improve sample efficiency.

Applied to the Synchromodal Transport Problems (STP), HRL involves:

- A high-level agent is deciding whether to re-plan an order (a simple binary yes/no decision).
- A low-level agent is performing the routing based on the decision how the order moves from origin to destination via available vehicles and paths.

The low-level routing subproblem forms its own MDP, with observations focused on local network details (vehicles, terminals, disruptions, timetables). Decisions occur at each terminal, determining the next vehicle or path until the destination is reached. Rewards can be extrinsic (environment-based) or intrinsic (goal-based).

This proceeding enhances scalability and transferability, as low-level agents can be reused across regions or network types. However, training stability remains a challenge because each policy's learning depends on others, creating non-stationarity, like MARL approaches.

3.3 Multi-Agent RL (MARL)

MARL differs from direct reinforcement learning by using multiple agents instead of a single global one. Each agent is responsible for decisions in a subset of the system (e.g., managing operations at one terminal or a small group of terminals). While each agent's Markov Decision Process (MDP) is smaller in scope, it ideally still accounts for relevant global network information to reduce partial observability [6].

This decentralized approach breaks the overall problem into smaller, more manageable local subproblems, shrinking the action space and potentially improving learning efficiency. However, it introduces new challenges [6–7]:

- Agents must adapt to each other's evolving policies, causing non-stationarity
- Coordination and cooperation between agents are difficult to achieve and often reduce sample efficiency
- Reward design becomes crucial to ensure shared objectives rather than competitive behavior

Overall, MARL is most suitable when the system can be divided into loosely connected parts or subregions. In highly interconnected (transport) networks with dependencies, its advantages diminish because complexity simply shifts from decision-making to agent interactions.

3.4 Hybrid RL to choose Heuristics

In the context of synchromodal transport planning (STP), hybrid reinforcement learning (HRL) approaches have emerged as a promising solution to balance optimality, explainability, and practical implementation effort. One such method is the use of RL to select among predefined heuristics, commonly referred to as RL-based Hyper Heuristics (RL-HH). This approach leverages the strengths of both RL and traditional heuristic methods, aiming to provide robust and adaptable decision-making in dynamic transport networks.

The core idea behind RL-HH is to restrict the RL agent's role to the selection of the most appropriate heuristic for a given subproblem, rather than having the agent directly solve the entire routing or scheduling task. When an order arrives at a terminal, the RL agent receives information about the current network state, including disruptions, available vehicles, and their capacities. Based on this observation, the agent decides whether re-planning is necessary and, if so, chooses one of several predefined heuristics to generate a new transport plan. This design keeps the agent's action space small and manageable, focusing its learning on the selection process rather than the full complexity of the transport problem.

A significant advantage of this approach is its generalizability. Since the set of heuristics is predefined and the agent's decisions are based on network state information, the same agent can be deployed across different terminals and even different networks, provided the underlying dynamics are consistent. This eliminates the need for training multiple agents, as required in multi-agent or hierarchical RL setups, and reduces both computational and technical overhead. Furthermore, the use of heuristics enhances the transparency and safety of the decision-making process, making it easier to explain and justify the agent's choices in operational environments.

However, the RL-HH approach does have limitations. By constraining the agent's search space to a fixed set of heuristics, the potential to discover globally optimal solutions is reduced. The agent may not be able to achieve the absolute best outcome, as the optimal policy might not be attainable within the available heuristics. Despite this, the practical benefits—such as reduced costs, improved explainability, and ease of implementation—often outweigh the loss in optimality, especially in real-world scenarios where transparency and adaptability are critical.

3.5 Comparison

Table 1 provides an overview based on strengths and weaknesses in various important aspects and a qualitative comparison of the introduced RL approaches in the context of STP. The different approaches shown in the table are compared based on four key criteria: explainability, transferability, and the required training and implementation effort.

Table 1: A qualitative comparison of RL-based approaches for STP

Property	Global RL	Regional MARL	Local MARL	HRL	Hybrid RL
Paradigm	Centralized	Distributed	Distributed	Hierarchical	Hybrid
Explainability	▼	▼	▼	▼	▲
Transferability	▼	–	–	▲	▲
Train. Effort	▲	▲	▲	▲	▼
Impl. Effort	–	▲	▲	▲	▼
Challenges	Scalability for larger problems; computational effort for training	Instationarity; computational effort for training; Defining appropriate regions; Reward Definition	Instationarity; computational effort; Reward Definition	Instationarity, Defining sub-tasks;complex implementa-tion;computational effort	Selecting appropriate heuristics to cover the search space
Suitable when...	network structure is static; computational resources are available;	network has well-defined weakly-connected sub-regions	network has terminals with similar properties across regions	network changes frequently; routing tasks are long/complex; emphasis on transferability	transferability, explainability are required;resources are constrained; optimal solutions are not crucial

Categories: low (▼), medium/neutral (–), high (▲) serving as a purely qualitative measure for relative comparison of the discussed approaches.

4. Our Methodology

In the STP, this subproblem is re-planning the transport of a single order under disruption, whereas the RL-agent's task is to decide if re-planning is necessary and, if so, to choose the appropriate heuristic in the current situation to react to the given circumstances. The action space is a discrete set of n different heuristics that can be applied to re-plan the transport and one action for sticking to the current plan (if still feasible). Once a heuristic is selected, it is applied to the underlying network and re-planning is performed accordingly. In the context of this problem, such heuristic is an arbitrary function that takes the current network state and an order as input and outputs a new transport plan for the order. Such heuristics can be simple rule-based algorithms (e.g., shortest path, earliest arrival, least cost) or more complex optimization algorithms and parametrized variants thereof.

4.1 Applied Heuristics

In the project context the following set of heuristics is applied, available to the agent and passed to the *RemunetGym* environment through the *heuristic_callables* dictionary, where they are used to compute heuristic results for agent-based heuristic decision-making:

- ***HeuristicCheapest***: This heuristic minimizes the cost of transport by evaluating the cost attributes of the (considered next) edge in the graph. This selects the route with the lowest monetary cost.
- ***HeuristicCo2***: This heuristic minimizes the CO_2 emissions by evaluating the CO_2 values of the edges in the graph. This heuristic selects the route with the least environmental impact.
- ***HeuristicSmallestDistanceOnRoad***: If the transport mode is ROAD, the cost is the distance of the edge. For other modes of transport, the cost is zero. This concept focuses on reducing road usage (for large distances), potentially for regulatory or efficiency reasons.
- ***DoNothingHeuristic***: Acts as a default or fallback heuristic and placeholder that performs no operation.
- ***HeuristicEarliestArrival:*** This heuristic minimizes the arrival time at the goal node. It prioritizes routes that ensure the earliest arrival at the destination. If the edge leads to the goal node, the cost is the *end_time* of the edge (otherwise the costs are calculated with 0).

Summarizing, the agent always chooses the most promising heuristic in the context of the problem setting during the RL optimization progress. The agent switches between the last selected heuristics from period to period. For example, it selects a cost-effective heuristic (if there is still plenty of time left) or an *earliestArrival* heuristic if there is less time left to meet the due date.

4.2 Agent rewards (Objective Functions)

The agent is evaluated ongoing throughout the whole RL-simulation process (see section 4.2, agent rewards). After each episode, the agent's policy is updated. At the end of the RL training process, the episodes are averaged (aggregated) and the policy is evaluated.

Rewards are numerical feedback signals that tell an agent how good or bad the outcome was of a taken action in each state. RL is based on the idea of learning from experience in a state-action-reward cycle, where an agent interacts with a given environment, taking actions based on its current state, receiving feedback (reward), and transitioning to new states. Over time, such an agent learns to optimize its action strategy (policy) to maximize its accumulated reward [4].

This interaction allows the agent to learn a decision-making strategy not just based on the current system dynamics, but also based on future, unknown events and the associated delayed reward.

But while RL can naturally address the complexity that is introduced due to the dynamic aspect of the problem, RL alone still struggles with the underlying complex, often combinatorial, routing action [5].

Agent rewards are generated after taking a certain action. These actions are always taken when the simulation is interrupted, and possible actions are processed in the decision data and evaluated in form of → simulation KPIs. The reward signal does not reflect direct consequences of the last action taken since the consequences of an action taken in a defined decision epoch may occur delayed [3].

The RL agent aims to deliver orders on time while minimizing emissions and costs, with its reward at each step reflecting the following balance between these objectives:

$$r_k = \begin{cases} -1 & if\ arrival\ at\ goal\ is\ too\ late \\ +1 & if\ arrival\ at\ goal\ is\ on\ time\ and\ both\ edges\ by\ train \\ +0.5 & if\ arrival\ at\ goal\ is\ on\ time\ and\ one\ edge\ by\ train \\ 0 & else \end{cases}$$

This formula favors rail transport over truck transport on the road. Costs and emissions are lower for rail transport than for truck transport anyway, when on-time delivery is ensured.

Each decision is triggered every time an order arrives at a terminal and if its capacity limits are exceeded (→ decision point, see chapter 5), with the initial decisions being made at the initial time t = 0. In the example shown in Figure 4, the observation space includes an order's current and goal position (i.e. $h1$, $h2$, $h3$) and the remaining time until its deadline.

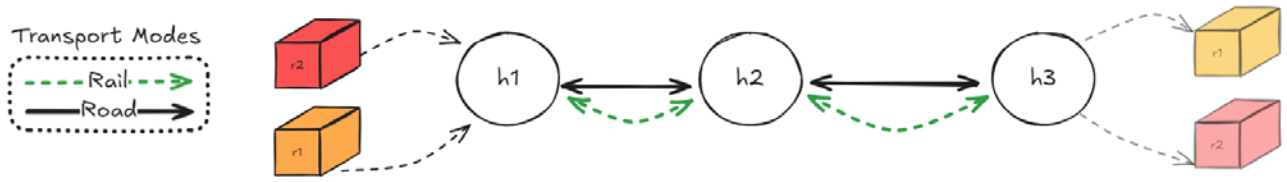

Figure 4: Transport network consisting of two requests, three terminals and two modes of transport (road and rail)

Figure 4 shows two railway connections from h1 to h2 and from h2 to h3 (featuring 100 time units travel time each) and the alternative road connections. Each railway connection (edge) is prone to delays occurring randomly, with a maximum of 50% of the travel time. In contrast, trucks are assumed to be available on demand at both terminals, providing faster and undisrupted travel times (e.g., 80 and 50 time units) for each edge. The resulting transport network in this example offers four possible transport options (combinations of truck and rail).

The corresponding available heuristics (see section 4.1) are either choosing the cost intense and fast option (truck, truck) or the cheaper and environmentally friendly option (train, train) which is slower and prone to delays. However, the agent learns to choose the right strategy based on its current location relative to the goal and the remaining time until the order's due time.

5. KPIs

The KPIs are a critical component for monitoring and optimizing the ongoing simulation's performance, ensuring that transport plans are both cost-effective and environmentally sustainable.

The *SimKpis* class is responsible for calculating and managing Key Performance Indicators (KPIs) for the defined simulation environment. It evaluates the performance of transport assignments in terms of feasibility, CO_2 emissions, transport costs, and delays for each decision point. The class is designed to dynamically update KPIs based on the simulation's progress and decision points.

The implementation of the class covers the following key concepts:

- KPI evaluation
- Dynamic updates
- Delay Management
- Feasibility Checks
- Aggregation
- Reusability

The following section details the implementation, including key components and core logic.

5.1 Implementation details

Within our implementation we designed a class *SimKpis* representing the aggregated KPIs for all orders. The class attributes are:

- *fixed_assignments:* Indicates if assignments are fixed.
- *fixed_costs*: Predefined fixed costs for different transport modes.
- *large_cost, large_co2*: Large values used for infeasible plans.
- *order_groups*: Groups of orders for evaluation.
- *first_call*: Tracks if this is the first evaluation.
- *last_results*: Stores the results of the last evaluation.

The total transport costs are calculated by adding the dynamic costs to the fixed costs (see Table 2 from [2]) depending on the individual mode that is used for travelling a certain edge.

Table 2: (Fixed) Costs for subcontracted transport depending on the mode

Transport Mode	Cost (per TEU)
Barge	0.14d
Rail	1.53 + 0.16d
Truck	76.4 + 1.04d

The decision point data (*dp_data*) is analyzed to check affected orders when an order arrives at a specific hub, and the simulation is interrupted at this decision point (hub).

The constructor initializes the class with simulation data (*sim_data*), graph data (*sim_graph*), decision point data (*dp_data*), and the current time. It analyzes the decision point data to check for affected orders. Finally, the simulation KPIs are evaluated and stored in a *results* object. These results for a single order are stored in an object of the *KPIResult* class. The attributes of the class comprehend:

- *order_id:* The ID of the order
- *plan_feasible*: Whether the plan for the order is feasible
- *current_delay*: The current delay for the order

- *total_predicted_delay*: The predicted delay for the order
- *co2*: The CO_2 emissions for the order
- *costs*: The transport costs for the order.

The methodological core is the *evaluateSimKPIs* function. It calculates Key Performance Indicators (KPIs) such as CO_2 emissions, transport costs, and delays for each order in the simulation. The logic includes the following six steps:

1. Initialization

- Initializes orderAssignment_feasible as True and an empty results list to store KPI results.
- Retrieves the *order_assignments* and their corresponding *timetable_entry_ids*. The timetable entries are extracted from the order assignments list.

2. Order Grouping: Groups order assignments by *order_id* into *SimKpis.order_groups* if not already grouped. This step is skipped if *fixed_assignments* is set to True.

3. KPI Calculation for Each Order

For each order_id in order_groups:

- Sorting: Groups are sorted by the start time of their edges in the graph.
- Initialization: variables for feasibility, total CO2, transport costs, and predicted delays.
- Edge Iteration within the graph:
  - Iterates through edges in the group and checks for scheduling conflicts with subsequent edges.
  - If a conflict is found (current edge ends after the next edge starts), the plan is marked *infeasible*, and large values are assigned for CO_2 and transport costs.
  - If no conflict exists, calculates CO_2, dynamic transport costs, and fixed transport costs for the edge.
  - Updates the total CO_2 and transport costs.
  - For the last edge, calculates the total predicted delay.
- Result Storage: Appends a KPIResult object for the currently inspected order, including feasibility, delays, CO_2, and costs.

4. Delay Calculation

- First Call: If this is the first evaluation and no affected orders exist, calculates delays for all orders using *evaluateSimDelays*.
- Affected Orders: If affected orders exist, calculate delays for them using *evaluateSimDelaysAffectedOrders*.
- Merging Delays: Merges delays from the current evaluation with delays from the last results.

5. Updating Results: Updates the *current_delay* attribute of each KPIResult based on the calculated delays, excluding infeasible orders.

6. Return Results: Logs the (intermediate) evaluation of the simulation KPIs and returns the list of *KPIResult* objects.

This method ensures that KPIs are dynamically updated based on the simulation's progress at each decision point, while handling feasibility and delays comprehensively and simultaneously.

6. Literature

[1] Y. Zhang *et al.*, “Synchromodal freight transport re-planning under service time uncertainty: An online model-assisted reinforcement learning,” *Transportation Research Part C: Emerging Technologies*, vol. 156, p. 104 355, Nov. 2023.

[2] Impact and relevance of transit disturbances on planning in intermodal container networks using disturbance cost analysis, Table A1 Costs for subcontracted transport: https://link.springer.com/article/10.1057/mel.2014.27/tables/9

[3] E. Pignatelli *et al.*, *A Survey of Temporal Credit Assignment in Deep Reinforcement Learning*; https://arxiv.org/abs/2312.01072v2, Dec. 2023

[4] R. S. Sutton *et al.*, *Reinforcement Learning, Second Edition: An Introduction*. MIT Press,

Nov. 2018.

[5] P. Dayan *et al.*, “Feudal Reinforcement Learning,” in *Advances in Neural Information*

Processing Systems, vol. 5, Morgan-Kaufmann, 1992.

[6] A. Wong *et al.*, “Deep multiagent reinforcement learning: Challenges and directions,” *Artif*

Intell Rev, vol. 56, no. 6, pp. 5023–5056, Jun. 2023.

[7] G. Papoudakis *et al.*, *Dealing with Non-Stationarity in Multi-Agent Deep Reinforcement*

Learning, Jun. 2019.

[8] T. Ambra *et al.*, “Towards freight transport system unification: Reviewing and combining the advancements in the physical internet and synchromodal transport research,” *International Journal of Production Research*, Mar. 2019.

The project

ReMuNet identifies and signals disruptive events and assesses their impact on multimodal transport corridors. It reacts quickly and seamlessly upon disruptive events in real-time. It supports TMS providers to improve route planning resilience. ReMuNet communicates alternative, pre-defined, multimodal transport routes to logistics operators and subsequently to truck drivers, locomotive drivers and barge captains. Through this, it enables a faster and adaptive multimodal network response. ReMuNet orchestrates route utilization, suggests transshipment points and optimizes capacity allocation, minimizing damage and shortening the recovery time. What is ReMuNet's core objective? As trailblazer for the Physical Internet, ReMuNet pursues the vision to enable and incentivize synchro-modal relay transport on European rail, road, and inland waterways to increase the holistic network resilience. It significantly reduces emissions and boosts freight transport corridor efficiency in case of disruptive events. Collaboration with stakeholders is essential to ensure Europe-wide practicability and acceptance.

Coordinator: FORSCHUNGSINSTITUT FUER RATIONALISIERUNG (FIR)

PARTNER		SHORT NAME
	FORSCHUNGSINSTITUT FUER RATIONALISIERUNG	FIR
	SVENSKA HANDELSHOGSKOLAN	HANKEN
	PTV PLANUNG TRANSPORT VERKEHR GmbH	PTV
	4PL INTERMODAL GMBH	INT
	MANSIO GMBH	MAN
	FRAUNHOFER AUSTRIA RESEARCH GMBH	FHA
	HAFEN WIEN GMBH	HWI
	WHITE RESEARCH SRL	WRE
	UNION INTERNATIONALE DES SOCIETES DE TRANSPORT COMBINE RAIL-ROUTE SCRL	UIR
	CONTARGO GMBH & CO KG	CON
	VEDIAFI OY	VED
	DANSK RODE KORS (DANISH RED CROSS)	DRC

FMI	ILMATIETEEN LAITOS	FMI
alice	ALLIANCE FOR LOGISTICS INNOVATION THROUGH COLLABORATION IN EUROPE	ETP-ALICE
SCHACHINGER LOGISTIK	SCHACHINGER IMMOBILIEN UND DIENSTLEISTUNGS GMBH & CO OG	SCH