Logistics for Uncertain Intermodal Distribution Networks

Tom Bürskens, Nicolas Garcia, Julian Komoßa, Raoul Luqué

July 2025

1 Introduction

The global automotive industry relies on extensive logistical operations to ensure that newly manufactured vehicles reach customers across continents in a timely and cost-effective manner. These operations involve intermodal transport stages between production plants, terminals, and dealerships. Efficient coordination minimizes costs and ensures timely deliveries, ultimately leading to customer satisfaction. This type of logistics planning is commonly referred to as the Auto-Carrier Transportation Problem and has been addressed in the literature, for example [TPD02]. Additionally, real-world planning must account for uncertainty, as transport resources may become partially or fully unavailable on short notice. To handle varying demand, companies often rely on contracts set up in advance as well as additional transport capacities that can be booked on short notice based on demand.

In this report, we formulate the described problem mathematically and motivate, develop, and discuss several heuristic approaches for solving it.

1.1 Mathematical Formulation

We begin by formally defining the deterministic version of the vehicle transportation problem, i.e., without uncertainty. The problem consists of a set of *locations* $L = P \dot{\cup} T \dot{\cup} D$, comprising plants P, terminals T, and dealerships D, a set of vehicles $V = v_1, \dots, v_k$, and a set of transports $A = a_1, \dots, a_n$. Additionally, there is a time horizon $\tau = [\tau_0, \tau_{end}]$ of days, which can be derived from the vehicle and transport data. Each vehicle $v \in V$ is associated with a starting plant $\text{start}(v) \in P$, a destination dealer $\text{dest}(v) \in D$, an availability date $\alpha(v) \in \tau$, and a due date $\beta(v) \in \tau$. Similarly, each transport $a \in A$ has a start location $\text{start}(a) \in L$, a destination location $\text{dest}(a) \in L$, and a departure and arrival date $\alpha(a), \beta(a) \in \tau$. Each transport also has a capacity $\kappa(a) \in \mathbb{Z}_+$ and a cost $\gamma(a) \in \mathbb{R}_+$.

The objective is to compute a minimum-cost feasible assignment $\sigma \subseteq V \times A$ of vehicles to transports and a delay indicator function $\delta : V \to 0, 1$. The delay function specifies whether a vehicle $v \in V$ is planned to be delayed ($\delta(v) = 1$) or not ($\delta(v) = 0$). An assignment is feasible if the capacity constraints of all transports are respected and, for each vehicle, the assigned transports form a valid path from its start to its destination. Specifically, no vehicle may depart before its availability date, nor be transported multiple times from the same location. The sequence of transports assigned to a vehicle must represent a path from its starting plant to its destination dealer. Additionally, we impose the restriction that a vehicle cannot be transported further on the same day it arrives at a location.

To evaluate the cost of an assignment, we define $\Delta(\sigma) = \{v \in V : \exists (v, a) \in \sigma : \text{dest}(a) = \text{dest}(v) \wedge \beta(a) > \text{dest}(v)\}$ as the set of vehicles that arrive with a delay. Let $\mu(v)$ denote the delay of vehicle v (0, if the vehicle arrives on time). We define

$\Delta_{planned} = \{v \in V \mid \delta(v) = 1\}$ and $\Delta_{unplanned} = \{v \in \Delta(\sigma) \mid \delta(v) = 0\}$ as the sets of vehicles with planned and unplanned delays, respectively.

The total cost of a solution is given by $c(\sigma, \delta) = c_\gamma + c_{planned} + c_{unplanned}$, where $c_\gamma = \sum_{a \in A} |\{v \in V \mid (v, a) \in \sigma\}| \cdot \frac{\gamma(a)}{\kappa(a)}$ is the proportionate cost of transport usage, $c_{planned} = |\Delta_{planned}| \cdot 200 + \sum_{v \in \Delta_{planned}} 50 \cdot \mu(v)$ is the penalty for planned delays, and $c_{unplanned} = |\Delta_{unplanned}| \cdot 500 + \sum_{v \in \Delta_{unplanned}} 100 \cdot \mu(v)$ is the penalty for unplanned delays.

We also consider the uncertain variant of this problem, where the capacities and costs of a transport a may vary on its scheduled day $\alpha(a)$. For brevity, we do not present a formal model for this setting here.

2 Heuristics

Even in the deterministic setting, the vehicle transportation problem introduced in Section 1.1 is $\mathcal{NP}$-hard, which can be shown by reduction from edge-disjoint paths in directed-acyclic-graphs. This computational complexity motivates the use of heuristic methods to solve large-scale, practical instances efficiently.

Each problem instance is defined by three types of input data. First, a set of locations. Second, a transport schedule over a given time horizon which provides all planned transports between these locations, along with their capacities and costs. Third, a set of vehicle orders that specifies when each vehicle becomes available at a plant and by when it must arrive at its destination. To evaluate and compare heuristics under uncertainty, we additionally consider ten realizations of transport schedules. These reflect possible deviations from the planned transports and allow us to assess the robustness of our methods.

To support understanding of the proposed heuristics, we introduce a simplified example instance, which will serve as a reference throughout this section. It is illustrated in Figure **??** and reflects the core structure of the vehicle transportation problem. Due to its simplicity, however, this instance is not representative of realistic problem settings.

2.1 Flow

The first heuristic we present models the problem as a multi-commodity integral minimum-cost flow problem (MCIMCFP). Specifically, we solve a flow-over-time problem on a static time-expanded graph, as described in [HHS07], focusing first on the deterministic case. In this approach, locations are represented as nodes and transports as edges, while vehicle orders (from start location l_{start} to destination l_{dest}) are modeled as flow demands. To incorporate temporal constraints, we construct a time-expanded graph with a node for each location on each day. Edges connect nodes according to available transports on the corresponding days. A single-commodity formulation cannot guarantee that vehicles reach their destinations on the correct day, so we define one commodity per group of vehicles that must arrive at the same location on the same day. Each commodity is assigned appropriate demands in the time-expanded network. Edge costs and capacities correspond to the cost and capacity of the respective transports.

The resulting flow network for the example introduced at the beginning of this section is illustrated in Figure **??**

Because the general MCIMCFP is $\mathcal{NP}$-hard to solve already for 2 commodities, see [EIS75], we developed a heuristic which solves multiple single-commodity flow problems instead, see section 2.1.1. Nonetheless, we also developed a heuristic which uses a mixed integer program to solve the entire MCIMCFP optimally, see section 2.1.2.

2.1.1 Solving Multiple Single-Commodity Flow Problems

In the deterministic case, the heuristic constructs the flow network using the realized data and processes the commodities in order of increasing due date. For each commodity, a single-commodity flow problem is solved, and the network capacities are updated to reflect the assignments extracted from the flow. The corresponding pseudocode is presented in algorithm **??**.

An exemplary solution of a single-commodity flow problem for the network depicted in figure **??** is shown in figure **??**.

In the uncertain setting, we iterate sequentially over all days, solving the MCIMCFP for each day based on the planned data. This way, we construct a planned assignment that assigns routes to vehicles scheduled for future transports. We then observe the realized transports for the current day and update the flow network accordingly, accounting for deviations from planned assignments. If a transport offers more capacity than anticipated, we assign as many vehicles as possible from those that were originally planned to use that specific route on a later day. The pseudocode for the uncertain setting is provided in algorithm **??**

2.1.2 Solving the Multi-Commodity Flow Problem optimally

Instead of solving multiple single-commodity flow problems, we also developed a heuristic which solves the MCIMCFP for each day optimally using a MIP.

2.2 Greedy

As a second heuristic, we implemented a greedy algorithm applicable to both the deterministic and uncertain setting. The algorithm iterates chronologically over all days and locations. For each day-location pair, it tracks the vehicles currently present and attempts to move them forward. The next location for each vehicle is determined by the shortest path (minimum number of locations) to its destination, where connections exist if there is any transport, regardless of frequency, between two locations. The greedy algorithm will try to send each vehicle to its next location using any transport that departs in that direction on that day, prioritizing cheaper transports and assigning vehicles with sooner due date first. If all transports to the next location are full, a vehicle remains at its current location. The corresponding pseudocode is provided in algorithm **??**.

In the uncertain setting, the algorithm first runs on the complete planned data on day one to identify vehicles that are planned to be delayed, marking their delay status as planned.

2.2.1 Greedy Algorithm based on Candidate Paths

This variant of the greedy algorithm modifies how a vehicle's next location is selected. Instead of following only the single shortest path to its destination, each vehicle maintains a list of short candidate paths, sorted by path length and average cost. These paths are considered sequentially if earlier options have no available capacity. An urgency function determines wether a vehicle is willing to use a more expensive option or prefers to wait another day. The corresponding pseudocode is shown in algorithm **??**.

2.3 Uncertainty

In the uncertain setting, the actual capacity of a transport may differ from its planned capacity, often resulting in reduced availability. To account for this variability, we adjust the planned capacity downward to better reflect likely realizations. The adjusted capacity for a given segment (defined by weekday, origin, destination, and truck number) is set such that it is lower than $p \in [0, 100]$ percent of the historical transports for the same segment. Assuming that future capacities are similar to those in the past, this approach ensures that in at least p percent of the cases, the adjusted capacity of a transport is less than or equal to its realized capacity, thereby increasing the robustness of the transport plan against capacity shortfalls.

3 Computational Evaluation

All evaluated heuristics were implemented in [Python] using [NetworkX] for network modeling and [Gurobi] via its Python interface to solve the MIPs. The full implementation can be found on GitHub[1]. The computations were conducted on the high-performance computing cluster CLAIX-2023 at RWTH Aachen University, specifically on nodes equipped with dual Intel Xeon 8468 Sapphire CPUs (48 cores per CPU) and 256 GB of memory. Only 4 of the total cores were utilized in our evaluations.

When evaluating a valid solution under the cost objective function, we accounted for horizon effects by only considering transport and vehicle delay costs of transports/vehicles that depart/become available more than seven days after the first vehicle and more than seven days before the final vehicle.

3.1 Results

We compare the four heuristics from Section 2 in terms of solution cost and runtime. Each heuristic was run once per dataset and realization. An overview of the logistic networks induced by the datasets is provided in Figure **??**.

Figure **??** shows the costs in the deterministic setting across all datasets. The x-axes represent the different realizations, while the y-axes indicate the respective costs. This layout will be used consistently in most of the subsequent plots.

It can be seen that the two flow-based heuristics consistently outperform the greedy heuristics. In particular, the flow and flow MIP approaches yield similar results. Note that in this context, the MIP based flow approach is not a heuristic but rather computes optimal solutions. The default greedy heuristic performs significantly worse, especially in comparison to the flow-based methods. We emphasize once more that the results on Dataset 1 should not be over interpreted due to the simplicity of the instance.

[1] https://github.com/RaoulLuque/MaHeuGroupProject

Figure **??** shows the comparison in the uncertain setting, with all values now normalized by the respective deterministic flow MIP solution. Similar analyses as for the deterministic setting can be made, with the exception that the the greedy candidate path heuristic got closer to the flow approaches. We note that in this setting the greedy heuristic was not able to deliver all vehicles for datasets 2,3,4.

Interestingly, all MIPs arising in the flow MIP heuristic were solved directly at the root node, without the need for branching. This suggests that, for the given instances, the LP relaxation consistently admits an integer optimal solution.

Figure **??** presents box plots of the relative cost difference between each heuristic's uncertain solution and the deterministic flow MIP solution. This measure can be interpreted as an indicator of the robustness of each heuristic with respect to unplanned changes in the transports across different realizations. The results indicate that the flow-based approaches, in particular the MIP algorithm, show the highest robustness.

We also want to compare the running times of the different heuristics. For this, consider the figures **??** and **??**, which show the running times in the deterministic and uncertain setting, respectively. Note that the y-axes are scaled logarithmically. As expected, the flow approaches take magnitudes longer than the greedy heuristics. In particular, the MIP based algorithm takes up to an hour per realization on dataset 1 in the uncertain setting.

At last, as discussed in 2.3, we tried to make our algorithms more robust by adapting the planned capacity according to historical data. In particular, we tested $p \in \{0.25, 0.5, 0.75, 1\}$, where the adjusted capacities equal the planned capacities for $p = 0$. Across all values, $p = 0.5$ yielded the best results. The difference of the costs for $p = 0$ and $p = 0.5$ can be seen in figure **??**. The results show that incorporating uncertainty handling does not lead to a consistent improvement or decline. Instead, the effect varies across heuristics and datasets.

Finally, due to space constraints, we do not provide a detailed analysis of why certain heuristics outperform others. However, preliminary observations suggest that the best performing heuristics tend to utilize more non-free transports, thereby reducing delay related costs.

4 Conclusion

In this report, we formulated a variant of the vehicle transportation problem under deterministic and uncertain conditions and developed four heuristic approaches: two flow-based (via successive single-commodity flows and a mixed-integer program) and two greedy strategies (plain and with candidate paths). Our computational study showed that flow-based methods consistently yield lower transport and delay costs. Notably, the non-MIP flow heuristic achieved solutions close to those of the MIP-based method, but with significantly reduced running times. The candidate-path greedy heuristic also delivered competitive results at a fraction of the runtime, making it a practical option for large-scale scenarios. In contrast, the plain greedy heuristic proved too simplistic and was unable to deliver all vehicles under uncertainty. As an outlook, extending the heuristics with local search techniques could further improve solution quality.