Dynamic Scheduling for Rail Freight: Reducing Extreme Departure Time Deviations through Demand Forecasting

A Dutch case study

Pim van der Velde

Delft University of Technology

Dynamic Scheduling for Rail Freight: Reducing Extreme Departure Time Deviations through Demand Forecasting

A Dutch case study

by

Pim van der Velde

Master thesis of the master: Transport, Infrastructure & Logistics
at the Technical University of Delft.

Student number:	4569857	
Thesis committee:	Dr. M.Y. (Yousef) Maknoon	TU Delft (TPM), Daily supervisor
	Dr. Y. (Yongqui) Zhu	TU Delft (CITG), supervisor
	L. (Luuk) van Egeraat	CGI, supervisor

Preface

Submitting this thesis marks the end of my master's in Transport, Infrastructure, and Logistics. The completion of this work would not have been possible without the support and guidance of several individuals, to whom I am deeply grateful.

Firstly, I would like to thank my daily supervisor, Dr. Yousef Maknoon, for his availability and invaluable insights. His expertise in operations research within the transport sector provided essential guidance throughout my thesis. I also extend my gratitude to my second supervisor, Dr. Yongqiu Zhu, for offering her perspective and helping me overcome challenges when I needed an additional viewpoint.

Secondly, I express my appreciation to Luuk van Egeraat from CGI for his dedicated support during our weekly update meetings. His availability to discuss and clarify complex topics proved crucial in navigating difficult moments in my research. I would also like to thank Sander Kapsenberg, whose presence during the monthly update meetings and thought-provoking questions consistently guided me in the right direction. I would also like to thank the company CGI for having the flexibility to invest all this time and effort in me and my thesis.

I would also like to thank my friends who are also doing their thesis, Frederiek, Coen, and Sabine. Their help, humor and the consistent shared coffee breaks provided a nice balance between work and relaxation. A special thanks to Frederiek for understanding my thesis and helping me brainstorm or discuss certain topics whenever I got stuck.

Lastly, I am deeply grateful to my roommates for their understanding and support, especially during challenging times. Their help with sometimes providing dinner more often during the busy periods, lightened my load significantly. Finally, a special thank you to my parents for their unwavering support, encouragement, and pride throughout my studies.

Pim van der Velde
Delft, January 2025

Executive Summary

This thesis investigates the development and evaluation of a dynamic scheduling model for rail freight in the Netherlands, aiming to improve scheduling by balancing forecasted network congestion and departure time deviations. The research focuses on analyzing ProRail's current scheduling limitations, constructing a deterministic demand forecasting model, and developing a mathematical model to optimize freight train scheduling. While the proposed approach shows promise in some scenarios, it does not consistently outperform ProRail's existing methods. This summary outlines the objectives, methodology, and key findings of the research.

The thesis begins by addressing the increasing demand for rail freight transport in the Netherlands and the challenges of meeting this demand while maintaining customer satisfaction. ProRail's current scheduling approach, which prioritizes requests on a first-come, first-served basis, often results in significant time deviations for later requests, harming customer trust. This research proposes a dynamic scheduling model that evaluates each request individually, determining whether to minimize the departure time deviation for the current request or account for forecasted congestion by offering an alternative departure time. The goal is to reduce extreme departure time deviations and distribute them more evenly across the planning horizon, while ensuring total deviations remain comparable. The primary research question guides the study:

"How can a dynamic scheduling model that balances minimizing forecasted congestion and departure time deviations per customer request, improve rail freight scheduling in the Netherlands?"

In the system overview, ProRail's existing scheduling practices and their limitations are analyzed. The manual, trial-and-error approach to freight train scheduling leads to suboptimal utilization of network capacity. Customers often reserve itineraries prematurely, due to uncertainty of receiving their preferred departure time, which results in cancellations and inefficiencies in the scheduling. This chapter highlights the importance of a more automated and customer-centric scheduling system that can improve operational efficiency and trust.

The thesis constructs a deterministic demand forecasting model to estimate section-specific network congestion. Using the C-Logit model, demand is distributed across paths based on factors like overlap distances. A mathematical scheduling model is developed, combining the dual objectives of minimizing departure time deviations and forecasted congestion. The weight parameter (w) enables trade-offs between these objectives, creating the base for the sensitivity analysis. Key assumptions, including uniform train speeds and simplified capacity constraints, are necessary to manage computational complexity but reduce real-world applicability.

The case study evaluates the dynamic scheduling model through experiments, a sensitivity analysis, and a repeated sensitivity analysis, aiming to identify a weight parameter (w) that consistently improves results compared to the baseline. The experiments tested the model under low, moderate, and high network capacities, exploring how different values of w influence the trade-off between minimizing departure time deviations and forecasted congestion. Sensitivity analysis further examined the impact of w on performance, while repeated sensitivity analysis randomized the order of requests to assess the model's robustness across different input scenarios.

Results showed that in low-capacity networks, smaller values of w could reduce extreme deviations but often increased total time deviations, presenting a trade-off question of what is desired overall by the customers. In high-capacity networks, the model's performance became insensitive to w, as requests were easily accommodated. The repeated sensitivity analysis highlighted significant variability in outcomes, especially in constrained networks, and no single value of w consistently and reliably improved results across all scenarios. These findings reveal that while the model can improve specific metrics under certain conditions, it lacks the robustness and reliability needed to outperform ProRail's current scheduling approach consistently.

The proposed scheduling model offers insights into balancing time deviations and network congestion but falls short of consistently improving results. The sensitivity of the results to weights, request order, and network conditions limits the model's reliability. While reducing extreme deviations is achievable, practical implementation challenges such as data limitations, pre-existing schedules, and the variability of real-world demand undermine its effectiveness. Nevertheless, the research contributes valuable knowledge on dynamic scheduling methodologies and congestion management.

The discussion situates the findings within the context of existing literature, emphasizing the novelty of integrating congestion and time deviation minimization without pricing mechanisms. Unlike previous studies focused on static timetables or dynamic pricing, this research introduces dynamic itinerary allocation to enhance operational flexibility. However, limitations, including simplified capacity assumptions, uniform train speeds, and reliance on deterministic demand forecasting, constrain the model's applicability. The research highlights the complexity of designing a robust scheduling system and identifies areas for further refinement.

The recommendations focus on enhancing ProRail's scheduling practices and future research directions. ProRail could improve demand forecasting tools to predict and manage future requests more accurately and explore the trade-offs between minimizing average and extreme time deviations to align with customer satisfaction goals. Refining congestion factor calculations and testing alternative penalty adjustments, such as exponential or scenario-specific scaling, could enhance scheduling decisions. Future studies should also incorporate dynamic adjustments to the weight parameter (w) using AI to optimize outcomes for varying request orders and network conditions. Addressing limitations such as uniform velocity assumptions, predefined paths, and simplified capacity constraints would improve the model's realism and applicability, enabling better alignment with ProRail's operational challenges and providing a foundation for scalable, customer-centric solutions.

This thesis underscores the potential of dynamic scheduling to improve rail freight operations but reveals the challenges of balancing competing objectives in a complex network. While the proposed model demonstrates some success in reducing extreme deviations, its inconsistent performance highlights the need for further research and refinement to achieve practical applicability and overall scheduling improvements.

Contents

List of Figures

List of Tables

1 Introduction

In a world driven by economic growth and globalization, the demand for freight transportation continues to surge (Lucía Morales, 2020; Statista, 2023). This increasing demand poses a challenge for transportation systems worldwide, especially in densely populated regions like the Netherlands. With projections indicating a staggering 50–93% increase in rail freight demand in the Netherlands by 2050 (Centraal Bureau voor de Statistiek, 2024; ProRail, 2021), the Dutch railway network is at a crossroads. How can the current infrastructure accommodate this growth while maintaining efficiency and reliability?

Among the various modes of freight transportation—road, rail, maritime, and air—rail transport stands out as one of the most sustainable options. It offers significantly lower emissions per tonne-kilometer compared to other modes, contributing to the global fight against climate change (European Environment Agency (EEA), 2023; Office of Rail and Road (ORR), 2023). As a society, making more use of rail freight is crucial not only for reducing environmental impact but also for alleviating highway congestion and fostering economic resilience.

Despite its advantages, the Dutch rail freight system is under pressure. ProRail, the Netherlands' infrastructure manager, faces the task of balancing the growing freight demands with a limited and highly utilized network. Even with annual investments of approximately €1.2 billion in infrastructure expansion (ProRail, 2020), the question remains: is expanding infrastructure alone sufficient to address the surge in demand? Or is a more dynamic and strategic scheduling approach needed?

This thesis seeks to address these challenges by exploring whether a dynamic scheduling model, designed to improve the current scheduling methods while meeting customer preferences, can offer a sustainable and efficient solution for rail freight logistics in the Netherlands.

1.1. Problem Statement

ProRail, the infrastructure manager of the Netherlands, currently adopts a first-come, first-served approach to handling freight requests (Infrasite, 2023). While straightforward, this method may not be best suited for managing the complexities of rising demand and peak freight periods. Requests are allocated as they arrive, filling preferred time slots early and leaving subsequent requests to face significant delays. This approach prioritizes early requests without considering the broader impact on the network or future requests, leading to inefficiencies and customer dissatisfaction.

Conversations with ProRail planners reveal that some freight requests experience departure time deviations ranging from multiple hours to several days from their preferred departure times. These deviations arise because the current system lacks the adaptability to balance network capacity utilization with meeting customer preferences. As the scheduling horizon progresses, customers submitting requests later face increasingly larger deviations from their preferred departure times, the question arises whether the current scheduling method can be improved, accounting for the dynamic and time-sensitive nature of rail freight logistics.

Despite ProRail's annual investment of approximately €1.2 billion in infrastructure expansion (ProRail, 2020), the challenge of managing rising freight volumes persists. Expanding infrastructure alone might not be enough to address the underlying inefficiencies of the scheduling process. This raises a critical question: could an improved dynamic scheduling approach, which considers forecasted demand to distribute departure time deviations more evenly throughout the scheduling horizon, help reduce the frequency and severity of large departure time deviations?

This thesis seeks to explore whether rescheduling earlier requests with small departure time deviations could reduce large time deviations (in size and quantity) for later requests. By developing a dynamic scheduling model that focuses on preventing forecasted congestion while meeting customer preferences, this research aims to improve the limitations of the current first-come, first-served approach. Drawing on insights from Kraft (2002), where smaller, distributed delays are found to be more acceptable to customers than larger, concentrated ones. This thesis investigates the effects of a dynamic scheduling system that offers alternative departure time deviations more evenly throughout the scheduling horizon, with a specific focus on reducing extreme deviations in size and quantity.

1.1.1. Request Handling Process

Understanding the current decision-making processes of ProRail for allocating freight requests onto the Dutch railway infrastructure is essential for identifying areas of improvement. ProRail manages a planning process that begins well before any freight requests are submitted, involving strategic, tactical and operational planning phases (ProRail, 2024a).

In the tactical planning phase, approximately 18 months prior to the timetable coming into effect, ProRail collaborates with freight customers to develop a base timetable. This phase involves calculating the maximum number of trains that can use specific tracks per hour, considering track capacity, station capacity, maintenance schedules and expected passenger and freight train operations.

During the tactical planning phase, from April to August, customers submit their freight train requests for the upcoming year. ProRail collects these requests and creates a first draft of the annual service timetable. Planners manually fit all requests into a conflict-free timetable, involving trial and error to accommodate as many requests as possible. The manual trial-and-error approach used for allocation is time-consuming and inefficient, limiting the ability to explore optimal solutions. This inflexibility also leads to underutilization of the network's capacity.

After the timetable is finalized, customers can still submit new requests or modifications, but acceptance depends on the remaining network capacity. ProRail's first-come, first-served system creates several issues. Customers who submit requests later face fewer options due to reduced network capacity, sometimes resulting in significant deviations from their preferred schedules. To avoid this uncertainty, customers tend to reserve itineraries early, even if they may not need them, leading to frequent cancellations or changes.

This section summarizes the key issues from the system analysis in Chapter 2, highlighting the problems ProRail experiences and leading into the need for a solution.

1.2. Actors & stakeholders

Understanding the stakeholders involved is essential to assessing the broader impact of this research. The stakeholders represent the groups or entities directly or indirectly influenced by the findings and proposed changes. These include ProRail and its customers.

The key actors involved in addressing the problem are:

Infrastructure Manager of the Netherlands (ProRail): ProRail is responsible for managing the railway infrastructure, optimizing track utilization, and planning train schedules. ProRail plays a crucial role in ensuring the efficient operation and capacity management of the rail network.

Customers: Customers include railway companies or other entities that operate or own freight trains and rely on the network for transportation. Their primary focus is on timely and efficient delivery of goods, often requiring coordination with other modes of transport.

Addressing the scheduling challenges discussed in Section 1.1 provides distinct benefits to these stakeholders. ProRail could optimize track utilization, reduce manual labor in scheduling, and foster trust among its customers, all within the constraints of the existing infrastructure. Customers, in turn, would experience greater alignment with their preferred time slots and paths, improving their satisfaction. However, increased flexibility might be required from customers submitting early requests, as they may need to adjust to rescheduling for the broader benefit.

The desired result of this research is to improve satisfaction for all stakeholders by designing and testing a new scheduling system that accounts for their needs and priorities.

1.3. Literature overview

To tackle the problem described in Section 1.1, literature research was conducted to critically examine current knowledge and limitations in the field, thereby identifying the research gap. This review draws on sources from databases such as Scopus, Google Scholar, Mendeley, and Scite, supplemented by snowballing methods to ensure comprehensive coverage. The central focus of this thesis is dynamic scheduling for rail freight transport, specifically leveraging demand prediction to decrease extreme departure time deviations.

The foundation of this work builds on concepts from revenue management, a field that has been extensively explored in passenger and freight transportation. Armstrong and Meissner (2010) conducted a detailed review of revenue management in railway planning, concluding that rail freight would benefit significantly from pricing optimization systems that learn from historical demand to predict future demand more accurately. While their study highlights the potential of these systems, it also identifies a significant research gap in network revenue management specifically tailored to freight railway systems. The general field of revenue management is important to take into account where the papers from Talluri and Ryzin (2006) and Gallego and Topaloglu (2019) are seen as the most detailed and important in this topic. Although revenue management traditionally emphasizes pricing and costs, its principles can be adapted for scheduling by assigning value or worth to itineraries and making decisions based on these values. This shift focuses on determining whether to allocate an itinerary immediately or reserve it for future requests, optimizing the overall use of network capacity.

In addition to revenue management, stochastic models have been employed to address allocation challenges in various industries. Van Slyke and Young (2000) discuss a stochastic knapsack problem that evaluates whether to accept or reject customer requests based on predefined criteria. While primarily applied to industries such as hospitality and aviation, this approach has potential in railway freight, where it can be adapted to evaluate whether offering an alternative itinerary is more beneficial than assigning a preferred option. Further insights are drawn from Kraft (2002), who examined dynamic car scheduling combined with stochastic train segment pricing to forecast demand, and also researched the effects of customer acceptance based on departure time deviations, thereby emphasizing the importance of minimizing such deviations. Although their study focuses on allocating cars to trains, the flexibility inherent in allocating trains to itineraries offers additional opportunities for dynamic reallocation. Unlike cars, trains can be rescheduled with minor departure time adjustments, enhancing the feasibility of real-time optimization.

Network allocation and capacity management are also critical aspects of this research. Li et al. (2024) explored train allocation to itineraries within a network, focusing primarily on revenue maximization through pricing. However, their approach does not consider offering alternative itineraries, which is central to this thesis. Meanwhile, integrated models combining operations planning and revenue management, as formulated by Crevier et al. (2012), provide a more holistic perspective. These models allocate trains and car blocks, yielding complex but accurate representations of network operations. While revenue remains their primary focus, the breadth of these studies offers valuable insights for developing more adaptable scheduling systems.

Path-based scheduling techniques further contribute to this thesis. Cacchiani and Toth (2012) developed models for single one-way lines, while Cacchiani et al. (2010) expanded this work to entire networks. These studies incorporate expected and already accepted demand, highlighting the trade-offs involved in balancing current requests with future expectations. The modeling of these trade-offs will be particularly useful for addressing the challenges identified in this thesis.

Despite the breadth of existing research, a significant knowledge gap remains in integrating demand prediction with dynamic scheduling to offer alternative itineraries. Current studies primarily focus on network revenue management or operational efficiency, with limited emphasis on how dynamic scheduling can be used to distribute departure time deviations more evenly across the scheduling horizon. Furthermore, the constraints of fixed pricing agreements in the Netherlands (Janssen, 2023) eliminate the feasibility of dynamic pricing as a tool to influence customer behavior. This thesis aims to bridge this gap by developing a dynamic scheduling model that processes requests sequentially, calculates the best itinerary to offer, and prioritizes reducing extreme departure time deviations in size and frequency. By providing alternative scheduling options tailored to forecasted demand, the proposed model seeks to enhance the utilization of the Dutch railway infrastructure while addressing the critical inefficiencies of the current scheduling approach.

1.3.1. Contribution Comparison

To provide a clear overview of the differences between the state-of-the-art knowledge presented in these papers and identify the research gap, table 1.1 is constructed. This table highlights the distinctions and demonstrates the unique contribution of this research.

Table 1.1: Comparison of literature against thesis topics

Papers / Topics	Dynamic Scheduling	Demand Prediction	Offering Alternatives	Reducing Extreme Deviations
(Armstrong & Meissner, 2010)		✓		
(Van Slyke & Young, 2000)		✓		
(Kraft, 2002)	✓	✓		✓
(Li et al., 2024)				
(Crevier et al., 2012)	✓			
(Cacchiani et al., 2010)	✓		✓	
(Cacchiani & Toth, 2012)	✓		✓	
This Study	✓	✓	✓	✓

While previous research has explored aspects of rail freight logistics such as stochastic optimization and capacity allocation, a significant gap remains in integrating demand forecasting with dynamic scheduling to reduce extreme departure time deviations. Studies such as Armstrong and Meissner (2010) and Kraft (2002) highlight the potential of demand prediction and minimizing departure time deviations, but do not address its application to dynamic scheduling decisions. Similarly, research on path-based scheduling by Cacchiani and Toth (2012) and Cacchiani et al. (2010) provides valuable insights but does not consider the sequential handling of requests or the distribution of time deviations across the scheduling horizon. Also, the stochastic models and optimization techniques discussed by Van Slyke and Young (2000) and the network allocation methods by Li et al. (2024) focus on different industries or do not consider offering alternative schedules to customers.

This research will bridge this gap by developing a dynamic scheduling model that integrates demand forecasting to determine the best itinerary to offer to each individual customer. The model focuses specifically on reducing extreme departure time deviations in size and frequency, within the existing railway infrastructure. Unlike previous work, this thesis prioritizes adapting the scheduling process to requests as they come in, while using forecasted demand to enhance the adaptability and efficiency of rail freight logistics.

1.4. Objective

The objective of this thesis is to contribute to the scientific understanding of rail freight scheduling by developing a dynamic scheduling model that focuses on reducing extreme departure time deviations in size and frequency. The model introduces a new approach by incorporating forecasted demand to distribute small deviations more evenly throughout the scheduling horizon with the goal of reducing the size and frequency of extreme departure time deviations at the end of the scheduling horizon.

Unlike previous research, which often emphasizes network revenue management or operational efficiency, this study specifically addresses the challenge of handling sequential requests in real-time while accounting for the dynamic nature of rail freight operations. By offering alternative itineraries based on forecasted demand, the model provides a fairer distribution of deviations, providing a structured method to handle congestion without introducing dynamic pricing or denying requests.

This research fills a critical gap in the literature by focusing on reducing extreme departure time deviations, an area that has not been thoroughly explored in rail freight logistics. The results aim to advance the theoretical foundation of demand-driven scheduling, providing insights that can be applied to improve the adaptability and reliability of freight rail networks.

1.5. Research questions

The main research question addressed in this thesis is:

"How can a dynamic scheduling model that balances minimizing forecasted congestion and departure time deviations per customer request, improve rail freight scheduling in the Netherlands?"

To answer the main research question, the following subquestions are formulated:

1. What are the limitations of the current scheduling approach used by ProRail, and how do they impact customer satisfaction?
2. How can a deterministic demand forecasting model be created using request data, and how can this information be integrated into the scheduling model?
3. How can a mathematical model be developed to dynamically schedule freight train requests while balancing the two objectives on forecasted congestion and departure time deviation?
4. How does the proposed dynamic scheduling model perform compared to the current approach?

1.6. Methodolgy

This section outlines the methodologies that will be used to develop a dynamic scheduling model for distributing rail freight requests. The methods are designed to address the research questions by integrating system analysis, data collection, mathematical modeling, and comparative analysis.

1.6.1. System Analysis and Interviews

What are the limitations of the current scheduling approach used by ProRail, and how do they impact customer satisfaction?

To answer the first subquestion, a detailed system analysis of ProRail's current scheduling processes will be conducted to understand the existing decision-making mechanisms and identify areas for improvement. This involves:

- **Interviews with ProRail Planners**: Engage with planners to gain insights into the entire planning process, including the structure of requests, timeline for the process, decision-making criteria and challenges encountered.
- **Mapping Current Processes**: Use information from the interviews to document the planning process, such as the overview of the planning process, freight train requests, request handling process, railway network and path allocation, decision-making by planners, problems, and opportunities for improvement.
- **Identifying Key Factors**: Analyze the collected information to determine useful factors affecting scheduling efficiency and customer satisfaction.

This method provides qualitative insights into the issues faced by ProRail and its customers, forming the foundation for accurately depicting the current scheduling process and from there develop an improved scheduling model.

1.6.2. Data Collection and Demand Forecasting

How can a deterministic demand forecasting model be created using request data, and how can this information be integrated into the scheduling model?

To answer the second subquestion, request data from customers, gathered over multiple months, for a specific day will be collected from ProRail. Focusing on essential elements such as origin, destination, and preferred departure times of freight train requests, the model remains manageable and allows for a clear demonstration of the scheduling approach's potential. Additional data will include:

- **Freight Paths**: Information on the predefined paths used by ProRail to plan requests within the network.
- **Distance Matrix**: Data to calculate distances between adjacent stations, aiding in travel time estimations.
- **Station Coordinates**: Geographical coordinates of stations for visualizing allocated orders.
- **Capacity network**: Capacity, per station and rail segment, defined in the amount of trains that can leave from a station to travel over a rail segment per hour.

A deterministic demand forecasting model will be developed using this data, which will be based on capacity and forecasted demand in the network. By using a combination of capacity and forecasted demand, a prediction can be made on whether there will be a high likelihood of congestion or not at certain sections in the network. By using a deterministic demand model, the approach simplifies demand prediction without the need for complex historical data analysis, which is beyond the scope of this thesis.

1.6.3. Mathematical Modeling and Literature Review

How can a mathematical model be developed to dynamically schedule freight train requests while balancing the two objectives on forecasted congestion and departure time deviation?

To address the third subquestion, this study begins by thoroughly analyzing ProRail's current scheduling system and translating its components into sets, parameters, and variables. Building on this analysis, a conceptual framework is designed to capture the goals of this thesis. The framework is then formalized into a mathematical model that dynamically schedules freight train requests while aiming to reduce extreme departure time deviations. The dynamic scheduling model processes requests sequentially, updating the model after each handled request. The development process includes:

1. **Problem Description**: Define the scheduling problem by specifying variables and parameters, simplifying the train scheduling model and network for mathematical modeling.
2. **Conceptual Model**: Outline the framework and objectives of the model, detailing how variables interact and setting the stage for mathematical formulation.
3. **Definition of Variables and Parameters**: Translate the conceptual framework into a structured mathematical format by specifying all sets, variables, and parameters.
4. **Formulation of Constraints and Objective Function**: Develop constraints representing operational limits such as capacities, train speeds and request fulfillment. Formulate a multi-objective optimization function to maximize network utilization and minimize time deviation per customer.
5. **Implementation and Solving**: Implement the model in Python using the Gurobi optimization engine. This involves coding the mathematical expressions, running the optimization algorithm and refining the model based on initial results.

An extensive literature review will support the model development, focusing on existing mathematical models in freight train scheduling and related fields. Insights from the literature will inform the selection of appropriate scheduling techniques, constraints, and objective functions, ensuring the model addresses the specific requirements of this research.

1.6.4. Comparative Analysis

How does the proposed dynamic scheduling model perform compared to the current approach?

To evaluate the performance of the proposed dynamic scheduling model, a comparative analysis will be conducted. This analysis will assess how the new model improves rail freight scheduling compared to ProRail's current first-come, first-served approach. The evaluation will focus on quantifying potential benefits and identifying improvements in handling rail freight requests.

Table 1.2: Key Performance Indicators for Comparative Analysis

KPI	**Description**
Cumulative Departure Time Deviation	The overall difference in departure time between preferred and actual scheduled departure times.
Cumulative Departure Time Deviation Squared	The overall difference in departure time squared between preferred and actual scheduled departure times. A decrease means there will be less extreme departure time deviations.
Average Cumulative Departure Time Deviation	For multiple experiments, gather the average overall difference in departure time between preferred and actual scheduled departure times.
Average Cumulative Departure Time Deviation Squared	For multiple experiments, gather the average overall difference in departure time squared between preferred and actual scheduled departure times.
Load Distribution	Analyzes how evenly requests are distributed across the network over time. Measured by how many trains are using the network at each point in time.
Average Departure Time Deviation per Request	The average departure time deviation per request across all experiments, providing an indicator of the average deviation experienced by requests.
Maximum Departure Time Deviation	The largest single departure time deviation observed across all requests, highlighting the most extreme delay experienced in the scheduling process.

2 System Analysis

Understanding the current decision-making processes of ProRail for allocating freight requests onto the Dutch railway infrastructure is essential for identifying areas of improvement. This chapter provides an analysis of the existing system, from the initial preparation of the network to the execution of train schedules. By examining each stage: network preparation, request handling, decision-making by planners, and the levels of decision-making, specific challenges and opportunities for improvement can be identified. This analysis sets the foundation for the problem description and the development of a more efficient scheduling model in later chapters.

2.1. Overview of the Planning Process

ProRail manages a planning process that begins well before any freight requests are submitted. The timeline for this process is illustrated in Figure 2.1, depicting the milestones from initial network evaluation to the day trains operate on the network.

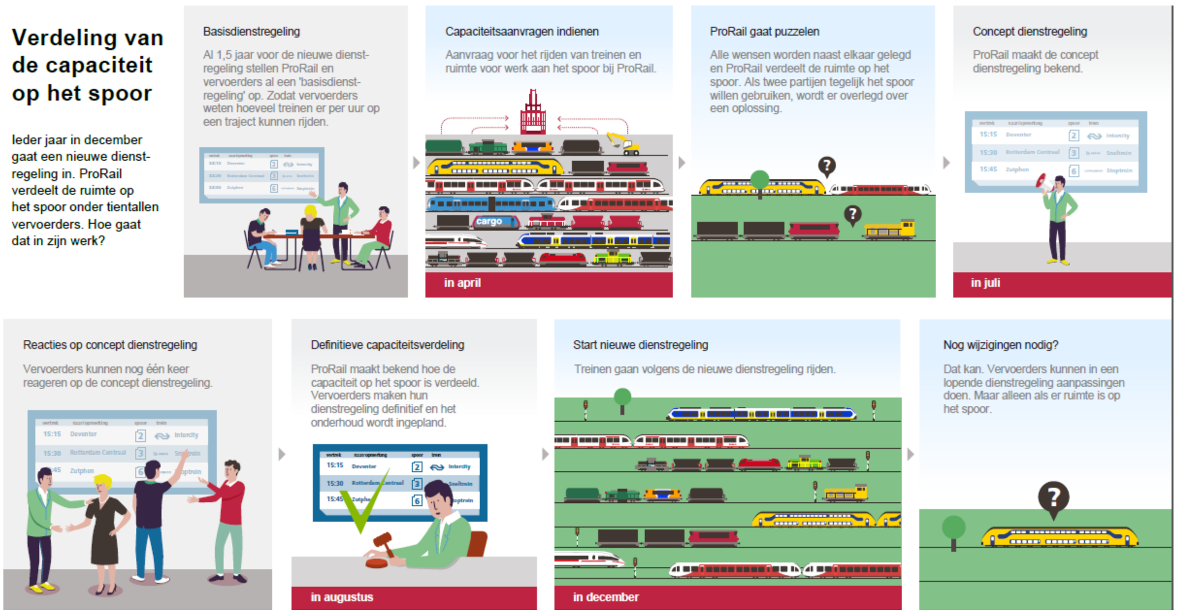

Verdeling van de capaciteit op het spoor
Ieder jaar in december gaat een nieuwe dienstregeling in. ProRail verdeelt de ruimte op het spoor onder tientallen vervoerders. Hoe gaat dat in zijn werk?
Basisdienstregeling
Al 1,5 jaar voor de nieuwe dienstregeling stellen ProRail en vervoerders al een 'basisdienstregeling' op. Zodat vervoerders weten hoeveel treinen er per uur op een traject kunnen rijden.
Capaciteitsaanvragen indienen
Aanvraag voor het rijden van treinen en ruimte voor werk aan het spoor bij ProRail.
cargo
in april
ProRail gaat puzzelen
Alle wensen worden naast elkaar gelegd en ProRail verdeelt de ruimte op het spoor. Als twee partijen tegelijk het spoor willen gebruiken, wordt er overlegd over een oplossing.
Concept dienstregeling
ProRail maakt de concept dienstregeling bekend.
15:15 Deventer 2 Intercity
15:30 Rotterdam Centraal 3 Sneltrein
15:45 Zutphen 6 Stoptrein
in juli
Reacties op concept dienstregeling
Vervoerders kunnen nog één keer reageren op de concept dienstregeling.
Definitieve capaciteitsverdeling
ProRail maakt bekend hoe de capaciteit op het spoor is verdeeld. Vervoerders maken hun dienstregeling definitief en het onderhoud wordt ingepland.
in augustus
Start nieuwe dienstregeling
Treinen gaan volgens de nieuwe dienstregeling rijden.
in december
Nog wijzigingen nodig?
Dat kan. Vervoerders kunnen in een lopende dienstregeling aanpassingen doen. Maar alleen als er ruimte is op het spoor.

Figure 2.1: Timeline of ProRail's Annual Service Planning (ProRail, 2024a)

2.1.1. First Planning Phase (18 Months Prior)

Approximately one and a half years before the annual service timetable comes into effect, ProRail collaborates with freight customers to develop a base timetable. This phase involves calculating and optimizing the maximum number of trains that can use specific tracks per hour, considering track capacity, maintenance schedules, and expected passenger train operations. This results in the formation of a set of predefined freight paths and an 'ideal' time-table which considers

2.1.2. Second Planning Phase (April to August)

Starting in April, customers submit their freight train requests for the upcoming year. ProRail collects these requests and creates a first draft of the annual service timetable. Between April and July, planners fit all requests into a conflict-free timetable. This process is manual and involves trial and error to accommodate as many requests as possible.

In July, the first concept of the timetable is presented to customers. While most requests are accommodated, some require adjustments, prompting further negotiations. From July to August, ProRail finalizes the timetable and incorporates additional requests or changes. During this period, ProRail responds to new requests faster due to the increasing rigidity of the timetable.

2.1.3. Third Planning Phase and Execution (Post-August)

After the timetable is finalized in August, customers can still submit new requests or modifications, but acceptance depends on the remaining network capacity. From December onward, the timetable is executed, and the freight trains operate according to the established schedule. Officially, new requests can be submitted up to two weeks before the desired departure date, but availability is limited by the existing capacity.

2.2. Freight Train Requests

Freight train requests submitted to ProRail vary based on customer needs. At a minimum, a request includes the origin and destination, departure date and time, and train characteristics such as length, weight, and maximum speed.

Customers can also specify additional details. Such as intermediate stations for loading or unloading, breaks or crew changes. Recurring schedules (e.g., every Monday, or the first day of each month) or a range of dates. Customers may also specify a time window for departure, such as the earliest and latest acceptable departure time, or they may define preferred arrival times at the destination. Furthermore, specific arrival and/or departure times can be provided for intermediate stations along the route, which could induce waiting times or enforce travel times to go from one station to the next, which leaves less flexibility for ProRail in the planning process. While these details allow customers to better align their logistics with ProRail's operations, they add significant complexity to the planning process. Implementing all these specifics is beyond the scope of this research, which focuses on developing a more generalized scheduling model. However, it is important to highlight these details so they can be considered in future research to further refine and improve scheduling systems.

2.3. Request Handling Process

The handling of freight train requests depends on the timing of the submission. Requests submitted between April and July must wait for the first draft of the annual service timetable, which is released in July, to see whether they have been accepted, modified, or denied. During this period, ProRail's planners focus on incorporating as many requests as possible without conflicts.

After the first draft is published, customers can submit new requests or modifications. ProRail usually responds within a day due to the timetable's relative stability at this stage. Accepted requests are fixed in the timetable and cannot be altered by ProRail without customer consent. However, the customer retains the freedom to modify or adjust the request, such as changing the departure time, origin, or destination, even after the timetable has been fixed. These changes can be accommodated if there is remaining capacity in the network.

Following the finalization of the timetable, new requests are only accommodated if sufficient capacity remains. Planners assess availability and offer alternatives if the preferred itinerary is not feasible.

2.4. Railway Network and Path Allocation

ProRail uses a predefined set of freight paths, developed during the first planning phase, which includes defined routes with specific departure times. These paths take into account the physical network, track capacities, station capacities, and ongoing rail work.

Each freight path specifies the sequence of stations, available departure times, and capacity constraints for a given time frame. For example, a path from Station A to Station D via Stations B and C might have departures at 09:10, 09:35, and 09:50, allowing three trains to use the path between 09:00 and 10:00.

In cases where a unique request requires a path that is not predefined, such as traveling from Station A to Station F via B, C and E, planners may merge existing itineraries to form a feasible route. For example, if two paths, A-B-C-D and G-C-E-F, share a common station (C), they can be merged to form a single itinerary from A to F. (For 60% of the requests which need to be allocated to the network, a newly formed unique path needs to be computed, see Section 7.3.)

Some tracks are bidirectional, allowing trains to travel in both directions. While this offers flexibility, careful coordination is needed to prevent conflicts.

2.5. Decision-Making by Planners

ProRail's planners play a key role in allocating requests, operating primarily on a tactical decision level. During the first stage, from April to July, planners manually adjust schedules to accommodate customer preferences (e.g. preferred departure time), often experimenting with different configurations to resolve conflicts.

Once the timetable is finalized, further changes can only be made if sufficient capacity remains. Planners offer alternatives when necessary, and customers are notified of small changes automatically. In cases of significant conflicts, planners call customers to discuss possible solutions, such as adjusting departure times or changing origin or destination stations.

Currently, ProRail does not maintain systematic data on the acceptance rates for rescheduled freight requests. This is difficult to store due to some rescheduling happening over the phone. Currently, when a request is changed, the original request gets canceled and a new request is made. However, with just information saying, request is canceled, from the data can not be concluded whether a new request is made from this, departure time is altered, path is changed or that the whole request is canceled with no alternative. From this limited data storage it is difficult to pinpoint how many request are deleted or simply altered.

2.6. Decision Levels in Scheduling

Time scheduling optimization can be performed at three different decision levels: strategic, tactical, and operational. This model focuses on the **tactical decision level**, which is appropriate for the medium-term planning horizon relevant to freight train scheduling:

- **Strategic Level**: Involves long-term planning over multiple months or years. Planners optimize the network to determine the maximum theoretical capacity, considering infrastructure investments and long-term demand trends. This level is addressed before requests can be submitted and is outside the scope of this model.
- **Tactical Level**: Pertains to medium-term planning, where planners allocate capacity to requests over the submission time horizon. This level focuses on balancing current requests with anticipated future demand, making decisions that affect scheduling over weeks or months.
- **Operational Level**: Deals with short-term, detailed planning, typically on a daily basis. It involves precise scheduling, including train movements, acceleration and deceleration, platform assignments at stations, and real-time adjustments. This level goes into too much detail for the scope of this thesis.

By focusing on the tactical level, the model aligns with the primary decision-making processes of ProRail's planners, allowing for relevant and practical insights into improving the scheduling system.

2.7. Problems

From July onwards, ProRail uses a first-come, first-served approach for requests being handed in. The result of this approach is that, customers who submit requests later will have fewer options due to reduced network capacity, which sometimes results in significant deviations from their preferred schedules. To avoid this uncertainty, customers tend to reserve itineraries early, even if they may not need them, leading to frequent cancellations or changes (some itineraries get changed up to 20 times). On top of this, the manual trial-and-error approach used for allocation is time-consuming and inefficient, further limiting the ability to explore optimal solutions, which may lead to underutilization of the network's full capacity.

2.8. Opportunities for Improvement

An opportunity for improvement, is to implement a dynamic scheduling model as an alternative for the first-come, first-served approach for the period after July. The idea here is to tactically offer early requests slight departure time deviations, to leave more room in the network for later requests. This approach would distribute small departure time deviations evenly throughout the planning horizon, rather than imposing large delays at the end to a few customers. By implementing dynamic scheduling, each request is handled individually before moving onto the next one which results in customers getting quick feedback on their implemented train schedule.

Demand forecasting could play a more prominent role in anticipating future demand and managing capacity accordingly. Automation of the scheduling process, would reduce manual labor and could increase efficiency, allowing for better network utilization and a higher customer satisfaction. Also, if customers trust the system's ability to provide their preferred departure times, they may be less inclined to reserve itineraries early out of a precaution. This could then lead to less cancellations or changes to the schedule.

In Chapter 4, the focus shifts to outlining the proposed system improvements. An analytical model is introduced to simulate the network, requests and decision making in a realistic yet simplified way. This approach aims to improve the system using mathematical methods, ensuring faster computation and easier model development, while still maintaining enough realism for this study. The model will be fully automated, with the human decision maker remaining external to the process.

3 Literature

This chapter presents a review of the existing literature relevant to dynamic rail freight scheduling, demand forecasting, congestion minimization, and customer satisfaction in the railway sector. The review highlights methodologies, findings, and research gaps that inform the objectives and methods of this thesis. A summary of the reviewed literature is presented in Table 3.1, followed by an analysis of key themes and the identified research gap.

Table 3.1: Summary of Literature Reviewed

Source	**Topic/Goal**	**Main Findings**	**Context**	**Model**	**Data**	**Uniqueness/ Transfer Attributes**
(Caprara et al., 2007)	Passenger Railway Optimization	Discusses optimization in passenger railway, lacking focus on cargo transportation.	European railways	Mathematical Models	Not specified	Shift from detailed planning to effective real-time control.
(Yue et al., 2024)	High-Speed Railway Timetabling	Developed heuristic based on Lagrangian relaxation for timetable optimization.	High-speed railways in China	Heuristic Algorithm	Beijing-Shanghai HSR data	Efficient in handling complex timetable challenges with a minimal duality gap.
(Brännlund et al., 1998)	Railway Timetabling	Novel optimization approach using Lagrangian relaxation.	Swedish railway	Integer Programming	Realistic Swedish rail example	Tested on passenger and freight trains, focuses on maximizing profit while respecting track capacities.
(Crainic et al., 1984)	Tactical Planning Model for Rail Freight	Optimizes routing, scheduling, and allocation of freight on rail networks.	Canadian National Railroads	Mixed Integer Multicommodity Flow	Canadian rail network data	Integrates several operational activities to develop global management strategies.
(Ghoseiri et al., 2004)	Multi-Objective Train Scheduling	Develops a model for passenger train scheduling optimizing fuel consumption and passenger-time.	Generic rail networks	Multi-Objective Optimization	Not specified	Uses ε-constraint method and distance-based optimization, not focused on freight.
(Yan & Goverde, 2019)	Train Timetabling and Line Planning	Combines LPP and TTP to optimize both line planning and timetabling.	Generic rail networks	Mixed Integer Linear Programming	Not specified	Focuses on robustness, regularity, and minimizing travel time for passengers.
(Caprara et al., 2006)	Real-World Train Timetabling	Addresses practical constraints in train timetabling using a Lagrangian heuristic.	Italian railways	Lagrangian Heuristic	Italian railway data	Incorporates real-world constraints like station capacities and maintenance operations.

Table 3.1: Summary of Literature Reviewed

Source	Topic/Goal	Main Findings	Context	Model	Data	Uniqueness/ Transfer Attributes
(Alaghband & Farhang Moghadam, 2019)	Scheduling Freight Trains on Single Tracks	Examines scheduling and allocation of freight, optimizing train travel time and freight tardiness.	Single line rail networks	Integer Linear Programming	Generated dataset	Heuristic algorithm designed to solve large-scale problems efficiently.
(Jovanovic & Harker, 1991)	Tactical Scheduling of Rail Operations	Presents a decision-support model (SCAN I) for the tactical scheduling of freight railroad traffic.	Rail operations	Simulation and Combinatorial Optimization	Major railroad examples	Introduced the SCAN system, enhancing decision-making in rail scheduling.
(Gallego & Topaloglu, 2019)	Revenue Management and Pricing Analytics	Provides an in-depth analysis of revenue management techniques and their applications across various industries.	Global	Revenue Management Models	Not specified	Extends revenue management principles to complex service industries including transportation.
(Talluri & Ryzin, 2006)	The Theory and Practice of Revenue Management	Discusses the comprehensive applications of revenue management, unifying theoretical and practical aspects.	Multisector	Theoretical Framework	Industry data	Bridges gap between theory and practice in revenue management, influencing multiple industries.
(Davis et al., 1987)	Optimal Capacity Expansion under Uncertainty	Introduces a mathematical model for capacity expansion with uncertain demand, incorporating stochastic control theory.	Capacity Expansion	Stochastic Control Theory	Not specified	Addresses the complexity of planning under uncertainty with numerical optimization methods.
(Armstrong & Meissner, 2010)	Revenue Management in Railways	Reviews gaps and opportunities in applying revenue management to rail freight.	Freight and passenger railways	Descriptive Review	Not specified	Highlights the lack of customer-focused dynamic scheduling.
(Li et al., 2024)	Network Revenue Management in Railways	Explores bi-level optimization for maximizing itinerary revenue through pricing mechanisms.	UK Rail Freight	Bi-Level Optimization	Industry data	Focus on profit-maximizing allocation of train itineraries.
(Crevier et al., 2012)	Integrated Operations and Revenue Management	Combines network planning with pricing strategies for rail freight.	Rail freight networks	Mixed Integer Linear Programming	Simulated data	Provides an integrated model for network yield and revenue optimization.
(Kraft, 2002)	Dynamic Car Scheduling	Combines dynamic scheduling and train segment pricing to forecast demand.	Rail freight networks	Multi-Commodity Network Flow	Simulated demand data	Introduces bid-price approaches for rail freight allocation.
(Feng et al., 2015)	Dynamic Model for Freight Overbooking	Uses Markov Decision Processes to manage overbooking in rail freight.	Chinese railways	Dynamic Programming	Simulated demand data	Focuses on balancing capacity allocation and demand variability.
(Cascetta et al., 1996)	C-Logit Route Choice Model	Improves path choice modeling by addressing overlapping paths.	Interurban road networks	C-Logit Model	Italian road network data	Enhanced realism in demand distribution for shared-path networks.

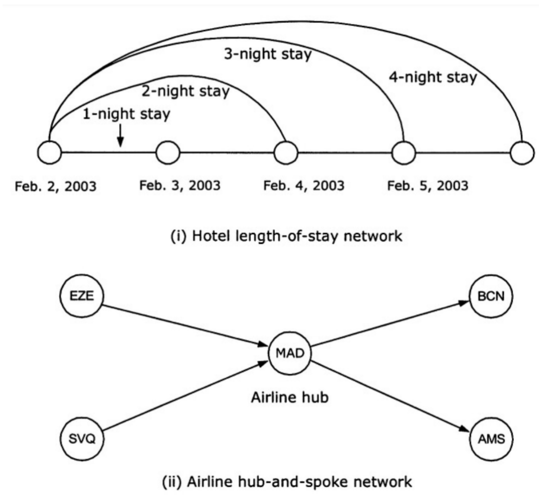
3.1. Revenue Management

Revenue management has traditionally emphasized maximizing profits through pricing strategies, as detailed in Armstrong and Meissner (2010), which provides an extensive review of revenue management in railways. However, these models often prioritize passenger services or rigid pricing structures over flexible freight operations. The seminal works of Talluri and Ryzin (2006) and Gallego and Topaloglu (2019) outline frameworks for maximizing resource utilization but rely heavily on price as the primary lever for optimization.

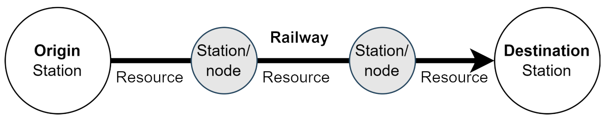
In the context of freight railways, Li et al. (2024) explores revenue management using a bi-level optimization model to maximize revenue through train itinerary allocation. While this approach effectively allocates resources, it lacks a customer-centric perspective that prioritizes minimizing extreme deviations in departure times. Similarly, Crevier et al. (2012) integrates operational planning and revenue management but remains focused on profitability rather than operational flexibility or customer satisfaction.

This thesis builds on the principles of network revenue management, reinterpreting them to prioritize customer satisfaction by offering alternatives when necessary, as opposed to solely maximizing revenue. By assigning intrinsic value to itineraries based on their alignment with customer preferences, this approach shifts the focus from pricing to operational efficiency and adaptability.

Figure 3.1 illustrates three examples of a hotel, airline, and railway network. All consisting of product and of the resources needed to form the product.



(a) Product & resource example hotel & airline (Talluri & Ryzin, 2006)



(b) Product & resource example railway network

Figure 3.1: Network revenue management visualization

3.1.1. Assortment optimization

Assortment optimization in revenue management involves the strategic selection of which products, in this context railway path time slots, to offer at any given time to maximize revenue. This may result in withholding certain products that could generate more revenue at a later time, rather than offering them at lower value moments. For example, during peak operational times, it may be beneficial to hold back certain slots on a freight path to better manage demand surges, thereby optimizing network utilization or revenue. Assortment optimization or management is an important topic for this thesis, for whether to offer an itinerary to a customer or withholding the itinerary for customers who hand in requests later.

3.1.2. Capacity management

Capacity management in railway operations involves planning and controlling the resources needed to meet both current and future demand, particularly under conditions of uncertainty. This encompasses decisions about when and how to expand infrastructure capacities, which must often be made amid fluctuating demand and limited resources. The study by Davis et al. (1987), which utilizes stochastic control theory to model capacity expansion, offers important insights into managing these challenges. It demonstrates how strategic investment decisions can be timed and scaled to optimize resource allocation efficiently.

For railway logistics, this means managing not just the physical tracks but also the allocation of time slots (the 'products' in this scenario), ensuring that capacity aligns with demand in both short-term operations and long-term infrastructure planning. By applying these principles, the railway infrastructure manager could improve their adaptability and responsiveness to changes in freight demand.

3.2. Stochastic Models and Dynamic Scheduling

Stochastic optimization has provided valuable tools for decision-making under uncertainty in various industries. The knapsack problem framework described by Van Slyke and Young (2000) offers insights into resource allocation by dynamically deciding whether to accept or reject incoming requests. Although primarily applied in sectors such as hospitality and airlines, this model highlights the potential of adapting stochastic methods to manage rail freight itineraries dynamically.

Kraft (2002) provides a pioneering approach by integrating dynamic car scheduling with train segment pricing. This model accounts for both accepted and forecasted demand, offering a framework that could be extended to manage freight requests on a network level. Similarly, Feng et al. (2015) applies Markov Decision Processes to overbooking in rail freight, illustrating how dynamic programming can optimize capacity under fluctuating demand.

The SCAN system introduced by Jovanovic and Harker (1991) further emphasizes the role of tactical scheduling in rail operations. By combining simulation with combinatorial optimization, SCAN supports interactive decision-making, showcasing how simulation-based models can enhance tactical freight train scheduling. While SCAN is focused on weekly or monthly scheduling, its methodology could inform real-time dynamic scheduling models, particularly in balancing conflicting objectives such as capacity constraints and customer satisfaction.

The current research adapts these stochastic frameworks to real-time freight scheduling, leveraging demand forecasting to predict bottlenecks and prioritize itineraries that minimize extreme deviations. This novel application addresses the limitations of static models in handling dynamic and uncertain demand patterns.

3.3. Network Allocation and Path-Based Scheduling

Path-based scheduling, as explored in Cacchiani and Toth (2012) and Cacchiani et al. (2010), provides a simplified yet effective approach to train scheduling, particularly in scenarios where fixed timetables must accommodate additional freight requests. Cacchiani et al. (2010) extends the problem from single-line to network-wide scheduling, incorporating existing passenger train timetables as constraints. While these models focus on maximizing network throughput, they do not integrate customer satisfaction metrics or real-time demand forecasting.

In contrast, Ghoseiri et al. (2004) introduces a multi-objective optimization model for train scheduling, balancing fuel consumption and passenger satisfaction by minimizing travel times. Although designed for passenger railways, this model demonstrates the feasibility of incorporating multiple objectives into scheduling frameworks. The e-constraint method used in this work could be adapted for freight operations to balance time deviations and congestion penalties, which are central to this thesis.

This thesis builds on these models by incorporating a deterministic demand forecasting framework, such as the C-Logit model outlined in Cascetta et al. (1996), to estimate section-specific congestion and improve path allocation dynamically. Unlike traditional methods that prioritize throughput, this approach balances operational efficiency with customer satisfaction by minimizing extreme time deviations.

4 Problem Description

Building upon the system analysis in Chapter 2, this chapter introduces the foundation for a new scheduling model for freight trains in the Netherlands.

Currently, requests submitted later in the scheduling horizon can face fewer options for allocation within the network, sometimes leading to significant departure time deviations. To address this, the proposed idea involves using forecasted future demand to tactically manage network capacity. By predicting where and for what time requests are likely to occur, itineraries with small deviations can be offered to early requests, preserving space in the network for anticipated future demand. This approach tries to keep capacity open for later requests and prevent possible congestion, reducing the risk of large deviations. The strategy makes use of the principle that it is better to offer a small time deviation to multiple customers compared to a large time deviation to a few customers (Kraft, 2002).

The goal of this research is to develop a scheduling model, that incorporates forecasted demand, with the objective of decreasing extreme departure time deviations without increasing the overall cumulative departure time deviations after scheduling all requests, compared to the current scheduling method. The model's decision-making involves determining which itinerary to offer a customer: either their preferred departure time or an alternative with a departure time deviation. The main challenge is deciding, for each incoming request, whether to prioritize reserving capacity for future requests by offering a deviating itinerary or to minimize the departure time deviation for the current request.

This section will also address the necessary simplifications made to the model in order to ensure that the thesis remains within a feasible scope while still maintaining its meaningfulness. The model is created at an analytical level, thereby reducing the complexities of the real world. This enables the model to illustrate potential improvements for the scheduling of freight trains without the need to simulate every detail of freight logistics.

The following sections will present the core structure of the scheduling model, focusing on the processes involved in request allocation, network capacity management, demand forecasting, graph representation and customer interactions.

4.1. Overview

For the dynamic scheduling model an infrastructure manager that owns and operates the railway network in the Netherlands is considered. The network is partially connected and consists of a set of nodes (stations) linked by railway infrastructure (arcs) (in section 4.2 the network graph is explained further). Customers of the infrastructure manager are railway companies or other entities that operate freight trains and wish to use the network to transport commodities.

The planning process involves two distinct phases: a submission period spanning several months and a single day of operation. During the submission period, customers submit freight train requests $r \in R$ sequentially, one at a time (batch requests are excluded), in the order $\tau_1 < \tau_2 < \cdots < \tau_R$. While the exact submission time is not relevant for this thesis, the order of the requests is important.

Each request corresponds to a single, unique train whose commodities cannot be combined with those of other trains. A request $r \in R$ is represented by a tuple τ_r containing information on the origin, destination and desired departure time t. The infrastructure manager processes these requests and allocates network capacity to the request of a customer but does not own the trains, the trains are owned by the customers.

The submission period ends with a cutoff point, after which no further requests can be made. The scheduled operations occur on a single day, referred to as the day of operation. Time within the day of operation is divided into two levels: hours ($h \in H$, where $H = 0, 1, \ldots, 23$) and minutes ($t \in T$, where $T = 0, 1, \ldots, 1439$). Minutes within an hour are indexed as $t = 60h + m$, where m ranges from 0 to 59. $t \in T_h$ are the minutes corresponding to hour h, where $T_h \subset T$.

The assumption is that the infrastructure manager has knowledge of future demand, expressed as the expected number of requests for origin-destination pairs for specific time intervals (e.g., an expected demand of five requests going from station A to station B departing at 12:00). This demand forecasting is based on historical data. However, in this thesis, demand will be established deterministically (see Chapter 7). While this introduces a certain level of simplification, it results in using 'perfect' forecasting, which is unrealistic. Perfect forecasting would raise questions about the trustworthiness of the results. To still keep a level of uncertainty in the forecasting, the demand model uses a deterministic number of requests per hour for an OD pair, which are then distributed across all possible paths connecting the origin and destination pairs. This way the model still uses a level of approximation, rather than a precise prediction. The detailed approach and calculation for forecasting demand is further explained in Section 5.1.

This goal of this thesis is to develop a dynamic scheduling model that processes requests sequentially, updating the network state after a request has been scheduled. Forecasted demand is used to predict how busy sections of the network are likely to become, allowing the model to strategically offer alternatives with small time deviations to requests early in the horizon. This preserves flexibility in the network for later requests, with the idea of reducing the likelihood of large departure time deviations. The objective is to improve customer satisfaction by minimizing large deviations for late requests while making sure the total time deviation (after allocating all requests) does not increase compared to the baseline scenario.

4.2. Graph Representation

The graph represents the railway network of the Netherlands. The railway network is represented using a time-space directed graph $G = (N_t, S, I)$, which models train movements and network utilization over discrete time steps. This approach defines a non-cyclic graph representing stations, arcs, and paths in both space and time. Accepted itineraries are allocated on this graph, representing paths from origin to destination with departure and arrival times at all nodes along the way.

- **Nodes (N) and Time-Expanded Nodes (N_t)**: N is the set of all stations in the network. $N_t = (n, t) | n \in N, t \in T$ represents each station at a specific time within the day. Each hour h relates to a set of 60 consecutive minutes $t = 60h + m$, where $m = 0, 1, \ldots, 59$.
- **Arcs (A) and Sections (S)**: A is the set of physical links between stations. Sections are time-dependent arcs, defined as $S = \{(a, t) \,|\, a \in A, t \in T\}$, representing the specific departure time from the first node of the arc. The sections S can be evaluated and managed on both an hourly basis (h) and a minute-level basis (t).
- **Paths (P) and Itineraries (I)**: P is the set of predefined paths, which are a combination of nodes $n \in N$ and arcs $a \in A$ without considering time. Each path $p \in P$ becomes time-dependent by assigning a departure time t, forming itineraries $I = \{(p, t) \,|\, p \in P, t \in T\}$. Itineraries consist of both a path and a specific departure time. Itineraries can be used interchangeably with a path + departure time combination. Where each itinerary $i \in I$ has a link to a path and departure time, expressed as: (p_i, t_i).

Each section $s \in S$ has a headway time h_s, representing the minimum time interval required between two trains departing from a time-expanded node over that section. This capacity constraint ensures that safety and operational requirements are met.

4.3. Dynamics of System and Offer Strategy

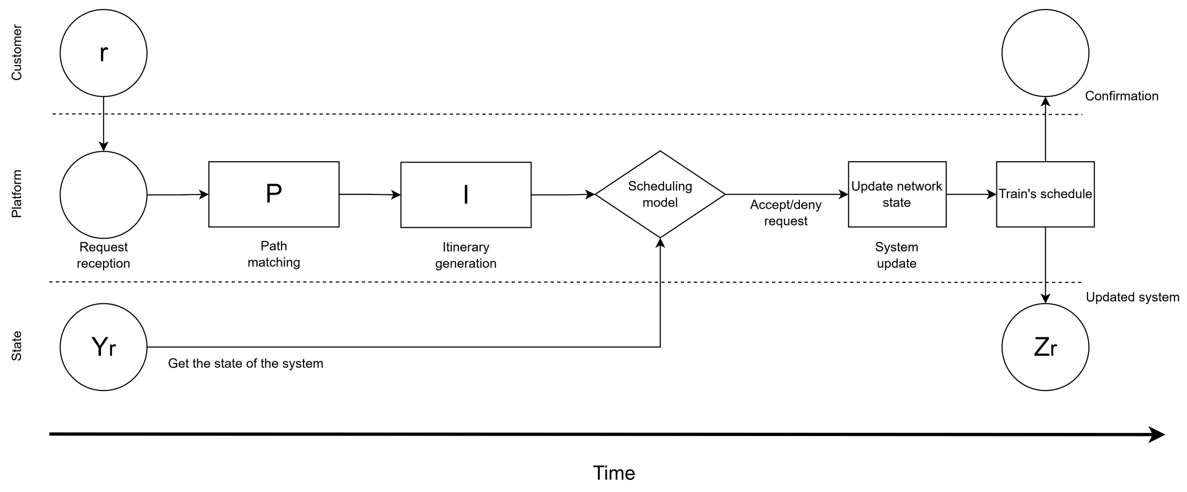


Figure 4.1: Request handling process

Figure 4.1 illustrates the dynamic scheduling process of a single request. Upon the arrival of request r, the state of the network is represented by Y_r. The state of the network includes the current capacity status for all arcs $a \in A$ and hours $h \in H$ in the network, which is defined as the maximum number of requests which can be allocated to an arc a at hour h: $C_{a,h}$, minus the actual allocated number of requests to arc a at hour h when request r comes in (so at time Y_r), defined as: $U_{ra,h}$ $(C_{a,h} - U_{ra,h}$ Subsection 5.1.2). Additionally, the network state incorporates the forecasted future demand, which is a prediction of the left over expected number of freight train requests which will arrive on each arc a during the hour h during request r (at time Y_r), represented by $D_{ra,h}$ (see Subsection 6.1.4 for the explanation on how to compute this parameter). This estimate is crucial for making informed decisions about allocating requests to itineraries while considering potential future constraints. How the network capacity and forecasted demand are portrayed and calculated as a whole, is explained in Section 5.1. After the model accepts the request by allocating it to an itinerary in the network, the updated state of the network is represented by Z_r. The notation Z contains the same information as Y, but is used to distinguish the state before (Y_r) and after (Z_r) processing request r.

The model aims to determine the best option to present to each customer upon receiving a new request τ_r. The process involves:

1. **Path matching**: From the list of predefined paths $p \in P$ all feasible paths are identified that fulfill the request's requirements $p \in P_{o_r d_r}$, which means the origin and destination of the request are matched to a path which contains the same origin and destination. The set $P_{o_r d_r}$ is request dependent. This is explained in more detail in Subsection 5.2.
2. **Itinerary generation**: From the set of feasible paths, generate a set of itineraries $i \in I_r$ consisting of a path and departure time (p, t). The set I_r is request dependent. How to compute this set per request is shown in more detail in Subsection 6.1.1. Methods to reduce this set of itineraries to a manageable number which is computationally feasible are explained in Subsection 5.3.
3. **Offer decision**: From the set I_r, which already complies with the preferred origin and destination from the customer, the scheduling model decides whether to present the customer with:
   - **Exact match**: A single itinerary that matches the customer's preferred departure time. The customer must accept this itinerary.
   - **Offer alternative itinerary**: An single alternative itinerary is offered with a deviating departure time from the customers preference. The customer must accept this itinerary.

   The offered itinerary to the customer with request r, includes the following details: origin, destination, all intermediate stations, departure and arrival times of all stations. The offered itinerary satisfies capacity constraints and customer preferences regarding the origin and destination.

The mathematical formulation of the decision-making process of what to offer to the customer is presented in Chapter 6.

In this thesis, for simplicity, each customer is offered a single itinerary, which is assumed to be accepted. While offering multiple options or discussing alternatives with the customer would be more realistic, especially when deviations from the preferred departure time are necessary, this would introduce customer choice probabilities and additional uncertainties into the scheduling model. Similar research by Cacchiani et al. (2010) explores offering three alternatives, allowing customers to choose one. However, this thesis focuses on testing a scheduling algorithm by identifying what the best possible itinerary would be for each customer and evaluating those results. This approach ensures that the results of the experiment (Chapter 7) relate directly to the newly proposed scheduling model, without being influenced by customer choices or uncertainties.

Currently, ProRail does not store data on the acceptance rates for rescheduled freight requests, see Section 2.5. This would have been interesting to take into account to make a prediction on whether a customer would accept an alternative offer from their original request. Due to this lack of data, the choice is now made that customers will always accept an offered request.

4. **System update**: After the itinerary is offered, the system is updated (Z_r) for the entire network, reflecting changes in network utilization and future demand forecasting. How the network is updated is explained in more detail in Subsection 6.1.4.

The customer interaction is limited to two occasions, when the customer hands in the request and when an itinerary is offered to the customer. Once the request is allocated onto the network, and thus an itinerary is offered to the customer, neither party can change their decision or offer. The allocated itinerary is fixed in the schedule.

4.4. Scope and Limitations

To reduce the complexity of real-world scheduling while ensuring the research remains meaningful, several assumptions are made:

1. **Restricted paths**: The system only considers predefined, frequently used freight paths. These paths avoid detours and cyclic routes. This reduces computational complexity and reflects ProRail's current operations.
2. **Similar travel time**: All freight paths between the same origin-destination pairs are assumed to have similar distances and are treated as equally favorable from a customer perspective. In this thesis, the focus is exclusively on departure time deviation and its impact on customer satisfaction. Considering travel time differences would introduce a trade-off between travel time and departure time deviation from a customer satisfaction perspective, which is outside the scope of this study.
3. **Non-stop train movements**: Trains pass through intermediate stations without stopping for loading or unloading. Although including waiting times would add realism, it significantly increases the number of choices the system must evaluate, making the problem too computationally challenging for this study.
4. **Uniform train speeds**: All trains are assumed to travel at a fixed speed of 80 km/h, creating homogeneous paths. While variable speeds would introduce more realism by reflecting overtaking and differing travel times, it adds too much complexity for this research and is left open for future work.
5. **Empty initial network**: The network is assumed to be empty at the start of scheduling. Although in reality the network already contains pre-scheduled trains, excluding them still allows for a focused analysis of the objective of this research, where a change can still be measured between the newly proposed scheduling model versus the current scheduling method of the infrastructure manager. While the simplification of an empty initial network reduces accuracy portraying the real world scenario, the impact of a potential improvement remains valid.
6. **Single tracks**: Each arc between two stations represents a single railway track, regardless of the actual number of tracks in reality. Reverse arcs between the same two stations are treated independently as single tracks, and bi-directional tracks are not considered. While this simplifies

the railway network to an unrealistic scenario, this thesis focuses on comparing time deviations in the scheduling process between multiple scenarios, not on replicating real-life conditions.

4.5. Conclusion

This chapter established the foundation for a dynamic scheduling model aimed at improving the allocation of freight train requests in the Netherlands. The current scheduling method often leads to significant departure time deviations for requests submitted later in the planning horizon. By incorporating forecasted demand into the scheduling process, the proposed model seeks to mitigate these deviations by strategically reserving capacity for future requests while ensuring that total cumulative time deviations do not increase.

Key elements of the model were introduced, including the representation of the railway network as a time-space graph and the sequential request handling process. The model's ability to dynamically update the network state after each request, leveraging forecasted demand and capacity constraints, forms the core of its decision-making strategy. Simplifications such as restricted paths, uniform speeds, and a deterministic demand forecast were introduced to balance the model's analytical focus with computational feasibility.

These foundations provide a clear framework for analyzing whether the proposed approach can improve scheduling efficiency and customer satisfaction. These elements set the stage for the conceptual framework 5, mathematical formulation 6, and case study 7, of the proposed scheduling approach.

5 Conceptual Model

Building upon the foundation laid in the problem description, this conceptual model elaborates on the key components of the proposed scheduling system. It shows how variables and parameters interact and provides preliminary mathematical formulations to illustrate implementation strategies. The sections follow the sequence structure depicted in Figure 5.1, showing the process of handling the requests of the customers. The scheduling process is split up in the state of the system (1 & 2), customer request (3), path matching (4), itinerary generation (4), objective function (4), allocating request (4 to 5) and finally updating the system state (5).

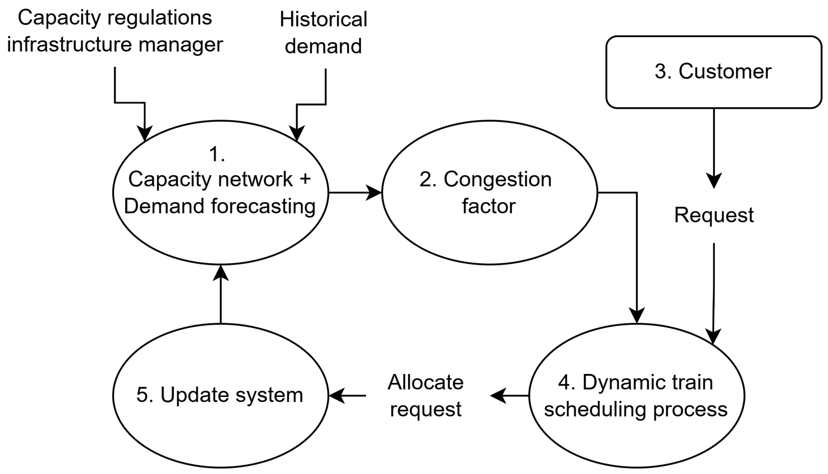

Capacity regulations infrastructure manager
Historical demand
3. Customer
1. Capacity network + Demand forecasting
2. Congestion factor
Request
5. Update system
Allocate request
4. Dynamic train scheduling process

Figure 5.1: Scheduling process

5.1. State of the system

The state of the system consists of two main elements: forecasted demand and network utilization. From these two elements, congestion factor values for all sections in the network can be calculated, which are needed for the objective function to predict how busy certain sections will become.

5.1.1. Forecasted Demand

Forecasting demand for freight trains presents a challenge due to the lack of information about the specific paths that trains will take between their origin and destination. While the number of requests per hour for each origin-destination (OD) pair is known, the distribution of these requests across the paths in the network is still to be decided. This is due to the possibility of multiple paths connecting the same OD pair, and these paths can also share overlapping sections, making it difficult to predict the demand for individual paths and their sections accurately.

To address this issue, this subsection introduces the Commonality Factor Logit (C-Logit) model, which adjusts demand allocation by accounting for path overlaps to ensure accurate demand estimation across the network and provide a robust framework for integrating these demand forecasts into the scheduling model.

Demand Distribution Problem
Distributing the forecasted demand for freight trains is challenging because only the origin and destination are known for each request per hour, while the specific paths taken remain unknown.

Figure 5.2 shows a simple railway network connecting an origin to a destination. Multiple paths can be used to travel from the origin to the destination. How can one make an accurate prediction on how to distribute the demand over the sections in the network?

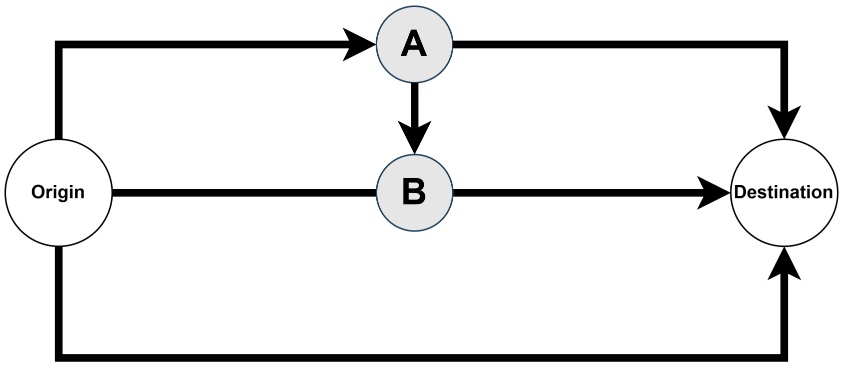


Figure 5.2: Railway network connecting Origin and Destination

Figure 5.3 illustrates the four possible paths for the network in 5.2. Note that some paths, such as the blue path, overlap with the yellow and red paths. This overlap complicates estimating the probability of requests being assigned to specific paths.

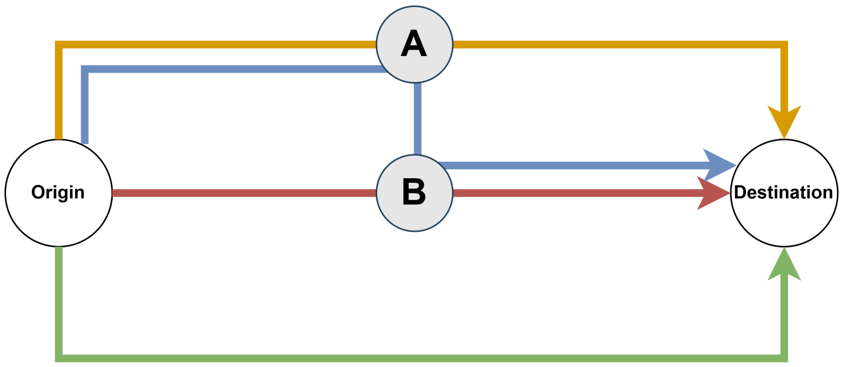


Figure 5.3: All possible paths connecting Origin and Destination

Overlapping paths create challenges in accurately distributing forecasted demand across the network. For example, if arc A-B did not exist, there would be three possible paths, and when all paths have equal utility (e.g. the same length), requests could be distributed equally. However, with overlapping paths, as shown in Figure 5.3, dividing demand equally over all possible paths would lead to an overestimation of demand on overlapping sections such as O-A and B-D. The Tables in 5.1 show the results of an equal distribution of demand across the paths and the results of what percentage of the requests would be allocated to the according sections.

To address this, an alternative method should consider the extent of overlap between paths and adjust the distribution accordingly. The Tables in 5.2 present a possibility for a more refined distribution, reducing requests sent over paths with significant overlap and increasing the amount of requests on paths with less overlap (these tables present an example of a possible improvement and are not based on any calculations).

Table 5.1: Distributing requests over paths using MNL

(a) Request distribution per path

Path	Requests
Yellow	25%
Blue	25%
Red	25%
Green	25%

(b) Request distribution per section

Section	Requests
O-A	50%
O-B	25%
O-D	25%
A-D	25%
A-B	25%
B-D	50%

Table 5.2: Improved request distribution accounting for path overlap

(a) Request distribution per path

Path	Requests
Yellow	25%
Blue	20%
Red	25%
Green	30%

(b) Request distribution per section

Section	Requests
O-A	45%
O-B	25%
O-D	30%
A-D	25%
A-B	20%
B-D	45%

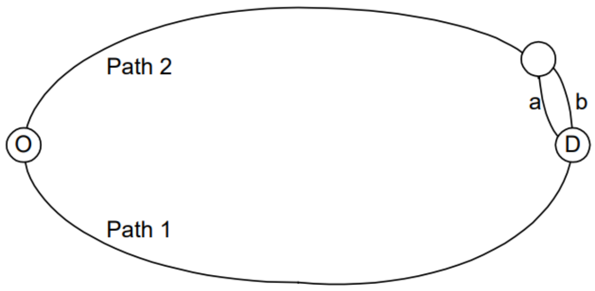


Figure 5.4: Path overlap example used by Ben-Akiva and Bierlaire (1999)

Another example showing the problem with the incorrect handling of path overlap by using Multinomial Logit Model (MNL) is shown in Figure 5.4 and an example is given based on the paper by Ben-Akiva and Bierlaire (1999). When the probability of choosing an alternative i from a choice set $\{1, 2a, 2b\}$ is calculated as:

$$P(i|\{1,2a,2b\}) = \frac{e^{\mu T_i}}{\sum_{j \in \{1,2a,2b\}} e^{\mu T_j}}$$

where T is the travel time of alternative i and μ is a scale parameter. In this example for Multinomial Logit, if $U_1 = U_{2a} = U_{2b}$, the probabilities are equal:

$$P(1|\{1,2a,2b\}) = P(2a|\{1,2a,2b\}) = P(2b|\{1,2a,2b\}) = \frac{1}{3}.$$

This result is independent of δ, a parameter representing small utility differences. When δ is much smaller than the total travel time T, the probabilities intuitively should reflect the correlation between alternatives, such as $P(1) \approx 50\%$ and $P(2a) \approx P(2b) \approx 25\%$. However, the MNL model assumes independent and identically distributed random utilities, which is invalid for correlated alternatives (e.g., $2a$ and $2b$), leading to unintuitive results in such cases. To get more accurate results which account for path overlap, the C-Logit model is used.

C-Logit model
To distribute the forecasted demand between origin-destination (OD) pairs accurately across the network, the Commonality Factor Logit (C-Logit) model can be applied, as presented in the paper from Cascetta et al. (1996). The C-Logit model builds upon the Multinomial Logit (MNL) model, which assumes all options are independent and distributes demand equally among alternatives based on their utilities (as shown in Table 5.1). However, the MNL model fails to account for shared characteristics between alternatives, such as overlapping rail infrastructure. The C-Logit model addresses this issue by incorporating a commonality factor, which penalizes paths that share significant rail infrastructure. This concept is based on the Nested Logit model (Ben-Akiva & Bierlaire, 1999), where alternatives with shared attributes are grouped to reflect correlations. By applying this principle specifically to overlapping paths, the C-Logit model provides a more realistic prediction of the demand distribution in the network and avoids over- or underestimating sections.

The forecasted future demand is based on historical data and provided as the expected number of requests for each OD pair with departure time t, denoted as $D_{od,t}$. The goal is to approximate the demand per arc per hour $D_{a,h}$ in the network by distributing the OD demand across the network using the C-Logit model.

Accounting for Overlap
To accurately represent the shared usage of infrastructure between different paths, the overlap is measured in terms of the total distance shared by two paths (Cascetta et al., 1996).

To distribute the OD demand across the network sections using the C-Logit model, the following steps are taken:

1. **Identify Feasible Paths:** For each OD pair (o,d), determine the set of feasible paths P_{od} connecting origin and destination $p \in P_{od}$. This path matching process is elaborated on further in Section 5.2.
2. **Calculate Commonality Factors:** For each path $p \in P_{od}$, calculate the commonality factor C_p:

$$C_p = \ln\left(\sum_{\substack{q \in P_{od} \\ q \neq p}} \left(\frac{L_{pq}}{\sqrt{L_p \cdot L_q}}\right)\right) \quad \forall p \in P_{od} \tag{5.1}$$

   where:

   - L_p is the total length of path p.
   - L_q is the total length of path q.
   - L_{pq} is the length of the overlapping arcs between paths p and q.
3. **Adjust Utilities:** Calculate the adjusted utility U_p for each path p:

$$U_p = V_p - \theta \cdot C_p \quad \forall p \in P_{od} \tag{5.2}$$

   where:

   - V_p is the observed utility of path p. Since travel times are similar across paths, V_p can be assumed to be zero.
   - θ is a scaling parameter reflecting the influence of the commonality factor (set to 1 in this context).

4. **Compute Path Probabilities:** Determine the probability $\mathbb{P}_p$ of assigning demand to path p:

$$\mathbb{P}_p = \frac{\exp(U_p)}{\sum_{k \in P_{od}} \exp(U_k)} \tag{5.3}$$

5. **Distribute Demand Across Paths:** Allocate the OD demand $D_{od,t}$ with departure time t among the paths $p \in P_{od}$ based on the calculated probabilities:

$$D_{p,t} = D_{od,t} \times \mathbb{P}_p \quad \forall p \in P_{od}, t \in T \tag{5.4}$$

6. **Linking Arcs to Itineraries** To streamline the representation of itineraries and their interaction with arcs in the network, the parameter $\delta_{ri,a,t}$ is introduced. This binary parameter indicates whether a specific itinerary i for request r uses arc a with departure time t. It is defined as:

$$\delta_{ri,a,t} = \begin{cases} 1 & \text{if itinerary } i \text{ for request } r \text{ departs on arc } a \text{ at time } t, \\ 0 & \text{otherwise.} \end{cases}$$

This parameter contains the temporal and spatial allocation of an itinerary, with its used arcs plus departure times, within the network, simplifying subsequent calculations. For any itinerary i, the sequence of arcs a it traverses and their corresponding departure times t are precomputed based on the path p and its departure time t_i (forming an itinerary i). This eliminates the need for recalculating which section is used in an itinerary, based on train velocity, for every step in the scheduling process.

The parameter $\delta_{ri,a,t}$ is calculated by determining the departure and arrival times along the arcs of a path, considering the constant velocity v of the train. For an itinerary i with a departure time t_i and path p, the travel time tt_a over an arc $a = (n_j, n_{j+1})$ is given by:

$$tt_a = \frac{d_{n_j,n_{j+1}}}{v},$$

where $d_{n_j,n_{j+1}}$ is the distance between nodes n_j and n_{j+1}. The train departs node n_j at t_{ri,n_j} and arrives at node n_{j+1} at:

$$t_{ri,n_{j+1}} = t_{ri,n_j} + tt_a.$$

For each arc a in the path, $\delta_{ri,a,t} = 1$ if t matches the departure time t_{ri,n_j}. This ensures $\delta_{ri,a,t}$ accurately represents the use of arc a at time t for the given itinerary.

Using the parameter $\delta_{ri,a,t}$, the demand $D_{a,t}$ for each arc a at time t can be calculated by summing the forecasted path demand $D_{p_i,t_i} \quad \forall i \in I$ over all itineraries i that contribute to the utilization of a at t. This is expressed as:

$$D_{a,t} = \sum_{i \in I} D_{p_i,t_i} \cdot \delta_{ri,a,t}, \forall a \in A, r \in R, t \in T \tag{5.5}$$

$$D_{a,h} = \sum_{t \in T_h} D_{a,t} \quad \forall a \in A, h \in H \tag{5.6}$$

where $\delta_{ri,a,t}$ determines whether itinerary i for request r uses arc a at time t. This formulation aggregates the contributions of all itineraries to the demand on a specific arc at a specific time, accounting for the allocation of forecasted demand across the network.

The demand values $D_{p,t}$ and $D_{a,t}$ represent the forecasted demand before any requests are allocated in the network. As requests are processed, this demand decreases. To track the remaining demand in the network after each request, the parameters $D_{rod,t}$, $D_{rp,t}$, and $D_{ra,t}$ are introduced. These represent the forecasted demand for each origin-destination pair, path, and arc, respectively, at departure time t in the network state Z_r.

Calculating Overlapping Distances
To determine the overlapping distance L_{pq} between two paths p and q:

- Represent each path as an ordered sequence of arcs and nodes, including the length of each arc between those nodes.
- Compare the sequences for paths p and q.
- Sum the lengths of the arcs that are common to both paths.

By doing so, the model accurately quantifies the extent of overlap, ensuring that the commonality factor reflects the true shared infrastructure length. This calculation will be used for the coding part in the case study in Subsection 7.3.9.

5.1.2. Network Capacity

Each section s or arc a has a minimum headway time h_s required between consecutive trains. When a train is scheduled over section s at time t, the section becomes unavailable for any other train departures during the interval $[t, t + h_s)$.

The theoretical maximum capacity $C_{a,h}$ of arc a during hour h is:

$$C_{a,h} = \left\lfloor \frac{60}{h_s} \right\rfloor$$

where h_s is in minutes. The actual utilization $U_{a,h}$ is the number of trains scheduled over arc a during hour h. $U_{a,h}$ represents the amount of requests which make use of arc a during hour h. This will be recalculated for the whole network for each arc a and hour h, after handling a request r. This is shown in Subsection 5.5 which is part of the system update. The calculation for $U_{a,h}$ is shown in 6.1.4. The remaining capacity for each arc a and hour h is represented by subtracting the amount of allocated requests from the maximum capacity $C_{a,h} - U_{a,h}$.

5.1.3. Section Congestion Factor

To identify which sections of the network will likely become congested, a congestion factor value $V_{a,h}$ is calculated for each arc a during hour h:

$$V_{a,h} = \frac{D_{a,h}}{C_{a,h} - U_{a,h} + \epsilon}, \quad \forall a \in A, h \in H$$

where ϵ is a small positive constant to avoid division by zero. $D_{a,h}$ is the accumulated demand for arc a during hour h, as calculated using the C-Logit model in Subsection 5.1.1. $C_{a,h}$ is the theoretical maximum capacity on arc a during hour h. $U_{a,h}$ is the amount of requests allocated to arc a during hour h.

By incorporating the C-Logit model into the demand distribution calculation, $V_{a,h}$ accurately reflects the expected congestion levels for each section, while considering the overlap of paths. However, this is just a prediction and does not reflect actual demand distribution.

The value $V_{a,h}$ helps to make a choice on how to schedule requests more evenly, in space and time, early on in the planning horizon. By identifying sections with a high value congestion factor (future expected busy), the system could focus on avoiding these sections. $V_{a,h}$ only represents arcs and hours, in equation 6.4 is shown how the congestion factor can be calculated for itineraries. This can then be used by the algorithm to prioritize scheduling on less congested paths (with a low congestion factor) by offering those as an alternative to customers.

5.1.4. Summarizing Parameters

The parameters for demand, maximum capacity, and used capacity are used often and may seem similar, but can sometimes mean something entirely different. Therefore additional clarification is needed to make sure no confusion arises. Demand parameters, such as $D_{od,t}$, represent the forecasted number of requests for an origin-destination pair with departure time t, while $D_{a,t}$ and $D_{a,h}$ reflect the forecasted number of requests on specific arcs a with departure time t or departing during hour h, respectively. $D_{p,t}$ represents the forecasted demand for path p departing at time t.

During the scheduling process, when scheduling request r (so before allocating the request r, at time Y_r), the leftover demand before allocating this request is denoted by $D_{rod,t}$, $D_{rp,t}$, $D_{ra,t}$ or $D_{ra,h}$, depending on the context. These parameters are mostly the same, but are request dependent and change during the scheduling process.

The maximum capacity, $C_{a,h}$, represents the total number of requests that can be allocated to an arc a during hour h, constrained by operational headway limits h_s . Used capacity, $U_{ra,h}$ represents the cumulative number of requests allocated to an arc during hour h, up to the arrival of request r. $U_{ra,h}(i)$ represents the cumulative number of requests allocated to an arc during hour h, if request r is allocated to itinerary i. $U_{ra,t}$, is a binary variable indicating whether arc a with departure time t is used or not at time Y_r.

The congestion factor parameters $V_{ra,h}$ and $V_{ra,h}(i)$ are used to quantify the level of congestion on specific arcs a during hour h when processing request r. The parameter $V_{ra,h}$ represents the congestion factor for arc a at hour h, considering all requests allocated up to request r. The parameter $V_{ra,h}(i)$ extends this by including the allocation of itinerary i to request r, reflecting how the allocation impacts the congestion level. To evaluate congestion for an entire itinerary, the congestion factors across all arcs and hours used by an itinerary are combined to calculate CF_{ri} (calculation shown in equation 6.4). This aggregated congestion factor, CF_{ri}, represents the cumulative congestion impact of assigning itinerary i to request r, taking into account all sections and time intervals involved in the itinerary.

To avoid confusion, it is essential to carefully observe the indices of these parameters: demand parameters are indexed by combinations of o, d, p, a, t, and h to specify origin-destination, path, arc, departure time, or hour; maximum capacity focuses on arc a per hour; used capacity involves request r to highlight its dynamic nature; and congestion factors add further granularity with respect to itineraries. These distinctions ensure accurate interpretation and application within the scheduling model.

To distinguish between these parameters, observe their indices carefully: parameters for demand often include indices such as o, d, p, a, t, and h, representing origin-destination pairs, paths, arcs, departure times, or hours, respectively. Parameters for maximum capacity use indices like a and h, referring to capacity on specific arcs per hour. Parameters for used capacity incorporate the index r, indicating their dynamic nature as they depend on the state of the network when processing request r. While the parameters may seem similar in form, their indices provide critical context to distinguish their specific meanings and roles within the scheduling model. These distinctions are essential to ensure clarity and avoid confusion when interpreting the network's state and scheduling decisions.

5.2. Path matching

For each request r, the system first attempts to match the origin and destination using the predefined freight paths provided by ProRail. In this research the solutions are dependent on the predefined freight paths provided by ProRail, creating new paths through a path optimization algorithm is beyond the scope of this study.

Each customer request r is represented by $\tau_r = (o_r, d_r, t_r)$, where:

- o_r: origin station.
- d_r: destination station.
- t_r: preferred departure time.

Feasible paths $P_{o_r d_r} \subset P$ are identified, where $P_{o_r d_r}$ is the set of predefined paths between o and d for request r. Each path $p \in P_{o_r d_r}$ is a sequence of arcs A and nodes N.

Additionally, the parameter $q_r(o, d, t)$ is introduced to represent whether request r contributes to the demand for a specific origin o, destination d, and departure time t. This binary parameter equals 1 if the request aligns with the given origin, destination, and time, and 0 otherwise.

The inclusion of $q_r(o, d, t)$ ensures a clear representation of how individual requests influence the demand model, aiding in the identification of feasible paths for a given request.

5.3. Itinerary generation

For each path $p \in P_{o_r d_r}$, itineraries are generated by varying the departure time t within acceptable limits (only options within the same day for this thesis). The set of departure times for each path is discretized into the time steps from that day. An itinerary $i \in I$ is feasible if all sections $s_i \in i$ in the itinerary are available, capacity wise, at the corresponding times, respecting previously allocated requests in the network with their corresponding headway times.

Important to mention is that there is a total set of itineraries $i \in I$, but per incoming request r the set of itineraries is also formed for that request noted by $i \in I_r$. This is explained further in Subsection 6.1.1. This set of itineraries is request dependent and will be used in the decision-making process to determine the best option to offer the customer. The scheduling model evaluates the itineraries in I_r based on the departure time deviation and forecasted congestion.

To reduce computational complexity for the case study, the focus there will be on key time points, such as hourly intervals. The process involves an outer for-loop over the hours h, followed by an inner loop that iterates over the finer time steps t (e.g., minutes) within each hour starting from the customer's preferred departure time. For each hour h, the system explores departure times $t \in \{(h-1) \times 60 + 1, \ldots, h \times 60\}$ to identify a feasible itinerary.

If a feasible itinerary is found at time step t in hour h, the system makes use of the assumption that the other itineraries within the same hour will have similar or worse characteristics. This assumption holds because the congestion factor values for arcs $V_{a,h}$ are calculated as an average per hour, making the values similar for every minute within that hour. Additionally, selecting the earliest feasible time within the hour minimizes the time deviation for the customer. Therefore, once a feasible itinerary is found, the system proceeds to the next hour $h+1$ without checking additional minutes for that hour. This approach ensures that all important feasible options are explored without making unnecessary calculations which would make the model computationally more complex.

5.4. Objective function

The objective function in this scheduling model seeks to balance two potentially conflicting goals: minimizing departure time deviations for customers and minimizing the congestion factor per request. These objectives may conflict because scheduling requests close to their preferred departure times could potentially increase congestion on certain network sections, especially during peak hours. On the other hand, prioritizing less congested paths may require deviating from preferred departure times, impacting customer satisfaction. Important to mention is these two objectives both serve the main goal of minimizing extreme departure time deviations. So when looking at all requests as a whole they both share the same objective.

5.4.1. Minimize Time Deviation

The first objective is to minimize the time difference between the preferred departure time t_r of request r and the scheduled departure time t_i of itinerary i. This ensures that customers experience minimal departure time deviations, improving their satisfaction with the scheduling process. The departure time deviation is expressed as:

$$\Delta t_{ir} = |t_i - t_r|$$

and one of the goals is to minimize Δt_{ir} for each request.

This does make use of the assumption that scheduling requests earlier or later to their preferred departure time is experienced as similarly preferable by the customer.

5.4.2. Minimize the Congestion Factor

The second objective focuses on reducing congestion on future expected heavily used sections. This is done by penalizing itineraries that include sections with high congestion factors. The congestion factor for an arc a during hour h, denoted as $V_{a,h}$, is a value of expected utilization based on forecasted demand and capacity. Higher values of $V_{a,h}$ indicate greater congestion, while lower values suggest that the section is less utilized.

For each request, the system evaluates itineraries and prioritizes those that use sections with lower $V_{a,h}$ values. This approach helps distribute demand more evenly across the network in space and time, preventing bottlenecks and preserving capacity for future requests. By reducing future congestion, more capacity is reserved in the network, which results in requests being handed in at the end of the time horizon to have more flexibility and potentially have smaller departure time deviations. This could improve the overall decrease of departure time deviation after allocating all requests.

The congestion factor is minimized by selecting itineraries that avoid sections with a high $V_{a,h}$ whenever possible, while still meeting operational constraints. How to calculate the congestion factor for the whole itinerary CF_{ri} with its corresponding sections is shown in 6.1.2.

5.4.3. Trade-off Between Objectives

The two objectives, minimizing time deviation and minimizing the congestion factor, are potentially conflicting. A balance between them must be achieved to improve the scheduling system's overall performance. Focusing too heavily on minimizing time deviations can lead to excessive congestion on key network sections, potentially causing substantial departure time deviations for later requests. On the other hand, focusing too much on minimizing congestion reduction might result in significant departure time deviations, negatively impacting customer satisfaction.

To manage this trade-off, a weighted parameter is introduced, allowing flexibility in determining their relative importance. The weighted objective function is defined as:

$$\text{Minimize } Z = w \cdot \text{Normalized Time Deviation} + (1 - w) \cdot \text{Congestion Factor},$$

where w represents the weights assigned to time deviation and the congestion factor. When $w = 1$ the focus is only on minimizing time deviation and if $w = 0$ the focus is only on minimizing the congestion factor. The mathematical model for the objective function is presented in Section 6.1.3.

To ensure the two objectives are comparable in the weighted objective function, the time deviation is normalized. The normalization is an approximation to bring the values in a similar range and is not based on any calculations:

- Time deviations Δt_{ir} are normalized by dividing the value by 120 minutes. This means a time deviation of 60 minutes corresponds to a normalized value of 0.5. This creates a scenario where the difference between two itineraries with a departure time deviation of 0 and a departure time deviation of 60 minutes becomes equal to 0.5 (when only taking departure time deviation into account).

  If the value for time deviation would be normalized the normal way, dividing it by the maximum time deviation, it would have to be divided by 1440 (the maximum amount of minutes in a day). This would always form an extremely small value for time deviation. Especially due to the fact that the time deviation will be close to zero most of the time. This would create a big unbalance in the objective and not produce desired results.
- Congestion factors are left as-is as a benchmark. This implies that, for example, when $w = 0.5$, the difference between the congestion factors of two itineraries must be greater than 1 for it to be favorable to offer the itinerary with a time deviation of 120 minutes with a less congested option.

Example of Trade-Off

To demonstrate the trade-off, consider two itineraries with the following properties:

- **Itinerary 1 (Path A):** Time deviation $\Delta t = 0$ minutes, congestion factor $CF = 1.5$.
- **Itinerary 2 (Path B):** Time deviation $\Delta t = 60$ minutes, congestion factor $CF = 0.5$.

The penalty calculation is defined as:

$$Z = w \cdot \frac{\Delta t}{120} + (1 - w) \cdot CF,$$

where w is the weight balancing time deviation and congestion.

Case 1: $w = 0.9$ **(high priority on time deviation)**

$$Z_{\text{Path A}} = 0.9 \cdot 0 + 0.1 \cdot 1.5 = 0.15, \quad Z_{\text{Path B}} = 0.9 \cdot \frac{60}{120} + 0.1 \cdot 0.5 = 0.50.$$

Path A is preferred.

Case 2: $w = 0.5$ **(balanced)**

$$Z_{\text{Path A}} = 0.5 \cdot 0 + 0.5 \cdot 1.5 = 0.75, \quad Z_{\text{Path B}} = 0.5 \cdot \frac{60}{120} + 0.5 \cdot 0.5 = 0.50.$$

Path B is preferred.

Case 3: $w = 0.1$ **(high priority on congestion)**

$$Z_{\text{Path A}} = 0.1 \cdot 0 + 0.9 \cdot 1.5 = 1.35, \quad Z_{\text{Path B}} = 0.1 \cdot \frac{60}{120} + 0.9 \cdot 0.5 = 0.50.$$

Path B is preferred.

Insights: For w close to 1, time deviation dominates. For w close to 0, congestion factor dominates. A balanced w results in a trade-off between the two objectives.

The goal of this normalization is to allow both objectives in the objective function to be in the same desired range of value, which is illustrated in Subsection 5.4.5. The weight parameter w can then be used to find the 'optimal' ratio between these two objectives by conducting a sensitivity analysis.

5.4.4. Time Sensitive Congestion Factor

The congestion factor is progressively scaled down for each request TF_r , placing greater emphasis on offering alternative itineraries earlier in the scheduling horizon. The rationale for implementing this time fraction is that, as the scheduling horizon progresses, actively offering alternatives becomes less critical. By this point, much of the network's capacity is already utilized, naturally increasing congestion factors on many arcs (see formula in Subsection 5.1.3). When nearing the end of the scheduling horizon, decisions are better focused on minimizing time deviations rather than attempting to tactically manage congestion.

For instance, consider a scenario where there is still enough capacity available on an arc for one train during a specific hour, and the forecasted demand predicts that two additional requests are still expected to be handed in for that arc. With deterministic forecasting, the model can reliably estimate the likelihood of these requests. If an alternative is offered to the first request, the second request can be accommodated with minimal deviations, as the forecasted demand is known with high accuracy. However, under uncertain forecasting conditions, the risks increase. If the second request never materializes, reserving capacity unnecessarily for it results in suboptimal utilization of the network. Scaling down the congestion factor penalty as requests are processed reduces the likelihood of such inefficiencies, ensuring that decisions are more focused on immediate needs as the horizon nears completion.

To implement this, a simple linear scaling function is used:

$$TF_r = 1 - \frac{r}{|R|}$$

Where r is the request being currently handled and $|R|$ is total amount of requests. This function ensures that the congestion factor penalty in the objective function has greater influence early on and decreases as scheduling progresses.

Besides linearly scaling down the congestion function penalty, other approaches could be considered. Such as an abrupt cutoff where the congestion factor is applied only for the first half of the requests and then completely ignored, or an exponential decay for a more rapid reduction and equal importance in the beginning of the scheduling horizon, these alternatives were not explored in this thesis and left for future research. Testing all these possibilities would require significant additional implementation and experimentation, which was beyond the scope of this thesis. Instead, this research focuses on implementing a simple linear scaling function or leaving it out to evaluate their relative effectiveness. The results of this comparison are presented in Chapter 7.

5.4.5. Sensitivity Analysis

A weighted sensitivity analysis explores the impact of varying w on scheduling decisions. By systematically adjusting w, the model identifies the optimal value w^* that achieves the best balance between minimizing time deviations and reducing congestion for a given set of requests R.

Illustrating the Trade-off Using the Objective Function

To understand how the objective function operates with the weights, consider the table below, which shows example values for time deviations and congestion factors for two itineraries along two paths.

Table 5.3: Example of departure time deviation and congestion factor for itineraries of two paths

Departure Time Deviation (minutes)	Congestion Factor (Path 1)	Congestion Factor (Path 2)
0	1.2	1.5
60	1.6	0.04
120	0.01	0.7
180	0.9	0.1
240	0.4	0.4

Using this table, consider the following example:

- For a departure time deviation of 60 minutes, Path 2 has a significantly lower congestion factor (0.04) compared to Path 1 (1.6). This makes Path 2 a logical choice, as the congestion factor heavily favors it, even with a slight deviation from the preferred departure time.
- Now, suppose the congestion factor for Path 2 with time deviation 60 increases from 0.04 to 0.8. The trade-off now becomes more complex, as the congestion factor is no longer extremely low. Here, the weight w in the objective function plays a critical role in determining what the scheduling model will prioritize. Either the itinerary with time deviation 0 on Path 1 with congestion factor 1.2, or the itinerary on path 2 with time deviation 60 and congestion factor 0.8, or maybe the itinerary with time deviation 120 on Path 1 with a congestion factor of 0.01.

In Section 5.4 mathematical formulation for the objective function is shown.

Parameter sensitivity analysis

A parameter sensitivity analysis offers an alternative to avoid the problem of finding the optimal w in the weighted sensitivity analysis. In this approach, the model minimizes only for the departure time deviation, while the congestion factor is treated as a constraint. For example only accepting itineraries which have a congestion factor below 1. This also eliminates the need for normalization, as the two objectives are handled separately.

However, this approach is too simplistic for this thesis. Unlike the weighted sensitivity analysis, which evaluates a trade-off between time deviation and congestion, the parameter sensitivity analysis forces the selection of itineraries based solely on one factor, ignoring the interplay between the two objectives.

Table 5.4: Example of departure time deviation and congestion factor for itineraries of two paths

Departure Time Deviation (minutes)	Congestion Factor (Path 1)	Congestion Factor (Path 2)
0	1.1	1.5
60	1.6	1.3
120	0.9	2.1
180	1.05	1.0
240	0.9	0.8

Take the example in Table 5.4. The paths connecting the origin and destination are popular and heavily used throughout the day. Most itineraries have a congestion factor around or above 1. If the constraint for the congestion factor forces the model to select an itinerary below 1, while minimizing for departure time deviation, it would choose the itinerary with a 120-minute deviation for Path 1. However, this is a substantial time deviation, and the congestion factor difference between the itineraries with a 0-minute and 120-minute deviation is relatively small. In this case, little is gained by offering such a large deviation.

If the weighted sensitivity analysis were applied to Table 5.4, it would likely choose the itinerary for Path 1 with a 0-minute time deviation and a congestion factor of 1.1. This approach considers the trade-off between time deviation and congestion, avoiding drastic decisions that prioritize one objective entirely.

For this reason, parameter sensitivity analysis is not pursued, and the weighted sensitivity analysis is preferred.

5.5. Updating the system

After an itinerary is offered to the customer, the system updates both the forecasted demand and network utilization to reflect the new state of the network.

The demand for the chosen OD pair and associated sections is reduced. Since the initial demand was equally divided among all feasible paths, the demand should also be subtracted from all these paths it was divided over. In Section 6.1.4 this is explained in more detail.

The sections which are used by the accepted itinerary are marked as occupied in the time-space graph $G = (N_t, S, I)$. Additionally, the section is marked as occupied not only during the scheduled time but also for the headway period h_s after, enforcing capacity constraints.

Sequential Processing The model processes each request r sequentially. After handling request r and allocating an itinerary onto the network, the network state is updated. With the next request $r + 1$ the whole process starts over again using the updated network state.

5.6. Conclusion

This chapter has developed a conceptual framework for the dynamic freight train scheduling system. It outlined the key components, including forecasting demand, managing network capacity, generating feasible itineraries, and formulating an objective function to balance minimizing time deviations and mitigating congestion. The use of advanced models, such as the C-Logit, ensures accurate demand distribution by accounting for path overlaps, while the system's sequential processing framework dynamically updates network states after each request.

By integrating these elements, the conceptual model lays a robust foundation for addressing the dual objectives of improving scheduling efficiency and enhancing customer satisfaction. The next steps involve implementing this framework in the mathematical model and validating its effectiveness through a case study.

6 Mathematical Model

This chapter presents the mathematical model developed to improve the dynamic scheduling of freight trains. The model sequentially processes incoming train requests, balancing customer preferences with network constraints and future demand. It integrates sets, parameters, and variables to represent the railway network and its time-dependent dynamics. The formulation captures the trade-offs between minimizing departure time deviations and mitigating network congestion per request, using an objective function and constraints designed to ensure efficient allocation of resources per request. Network state updates reflect the real-time impact of each scheduling decision, enabling the model to adapt to changing conditions throughout the scheduling process.

6.1. Mathematical Formulation

The mathematical model processes each freight train request sequentially. Each request is handled one at a time, updating the network state after processing each request. In table 6.1 all the sets, parameters and variables used for the scheduling model are presented.

6.1.1. Itinerary Generation

Freight train requests, denoted as $r \in R$, arrive sequentially, each represented as a tuple $\{o_r, d_r, t_r\}$, where $o_r \in N$ and $d_r \in N$ indicate the origin and destination stations, and $t_r \in T$ specifies the preferred departure time. Here, N represents all possible stations, and T consists of discrete time points within a day.

The railway network is modeled using a set of predefined paths, $p \in P$, each representing a sequence of nodes connecting an origin to a destination. In the set of predefined paths there are subsets which consist of all paths connecting an origin to a destination, noted by $P_{od} \subset P$. Formally, a path $p \in P$ is represented as $p = (n_1, n_2, \ldots, n_L)$, where each consecutive pair $(n_\ell, n_{\ell+1})$ corresponds to an arc $a_\ell \in A$. Given a request r, the subset of paths $P_{o_r,d_r} = \{p \in P : \text{origin of } p = o_r, \text{ destination of } p = d_r\}$ is identified, containing all paths satisfying the request's origin-destination pair.

To generate itineraries, each path $p \in P_{o_r,d_r}$ is paired with a departure time $t \in T$, creating a set of itineraries, $I_r = \{(p,t) \mid p \in P_{o_r,d_r}, t \in T\}$. These itineraries consider all possible combinations of paths and departure times that comply with the request. Each itinerary $(p_i, t_i) \in I_r$ specifies the path p_i and the departure time t_i from the origin o_r.

The following example shows this process. Suppose the network consists of stations $N = \{A, B, C, D\}$ with predefined paths $P = \{p_1 = (A, B, D), p_2 = (A, C, D)\}$. A request r with $\{o_r = A, d_r = D, t_r = 100\}$ leads to the paths $P_{o_r,d_r} = \{p_1, p_2\}$. If $T = \{0, 1, \ldots, 1439\}$, then itineraries for p_1 include $\{(p_1, 0), (p_1, 1), \ldots, (p_1, 1439)\}$, and similarly for p_2. Thus, I_r combines these sets, resulting in a comprehensive list of feasible options.

6.1.2. Congestion Factor Calculation

In Subsection 5.1.3, the formula for calculating the congestion factor for an arc a and hour h is introduced. This formula is used to determine the congestion factor before allocating requests to the network. However, as requests are allocated, the network's capacity usage increases and the forecasted demand decreases, necessitating updates to the congestion factor.

During the allocation process, it is smart to account for not only the current congestion factor up to request r (see Equation 6.17), but also the anticipated changes resulting from allocating itinerary i to request r. This requires calculating the congestion factor for every itinerary in the set I_r for request r, incorporating the capacity and demand across all arcs and hours in the network up to and including the current request.

Currently, $U_{ra,h}$ represents the number of requests allocated on arc a during hour h up to request r, i.e., before deciding how to handle request r. To represent usage after allocating request r to itinerary i, define:

$$U_{ra,h}(i) = U_{ra,h} + \sum_{t \in T_h} \delta_{ri,a,t}, \quad \forall h \in H, r \in R, a \in A, i \in I_r \tag{6.1}$$

where:

- $U_{ra,h}$ is the capacity usage in state Y_r (i.e., before request r is allocated, see Figure 4.1 for explanation on Y_r)
- $\delta_{ri,a,t} = 1$ if itinerary i uses arc a at departure time t
- T_h is the set of all minute-level departure times within hour h.

Hence, $U_{ra,h}(i)$ is the usage in the new state Z_r, if request r is allocated to itinerary i (see Figure 4.1 for explanation on state Y_r and state Z_r).

The forecasted demand is also updated when a request is allocated. This update is independent of the specific itinerary chosen and depends solely on whether the request is allocated. By increasing the request number by +1, the forecasted demand decreases accordingly for the corresponding origin-destination pair and departure time. The exact mathematical calculations on how to update the demand is shown in Subsection 6.1.4.

The congestion factor for arc a during hour h, for when itinerary i would be allocated to request r, is defined as:

$$V_{ra,h}(i) = \frac{D_{r+1,a,h}}{C_{a,h} - U_{ra,h}(i) + \epsilon}, \quad \forall h \in H, r \in R, a \in A, i \in I_r \tag{6.2}$$

where:

- $D_{r+1,a,h}$ is the forecasted demand for arc a at hour h in state Z_r
- $C_{a,h}$ is the capacity on arc a for hour h
- ϵ is a small constant to avoid division by zero

By including $\sum_{t \in T_h} \delta_{ri,a,t}$ inside $U_{ra,h}(i)$ and looking at the next request for $D_{r+1,a,h}$, the algorithm is effectively "thinking ahead," i.e., computing how congested arc a and hour h would be once request r has actually been assigned to itinerary i.

Once the congestion factor for all arcs is calculated, the congestion factor for an itinerary can be determined. Two methods are evaluated in this thesis, as presented in Chapter 7. These methods differ in whether the congestion factor per section is squared (Equation 6.3) or not (Equation 6.4). Squaring the congestion factor magnifies the effect of highly congested sections, ensuring that heavily used sections receive higher penalties compared to less congested ones. This approach, inspired by multi-commodity network flow (MCNF) problems, prevents the sum of many small penalties from overshadowing the impact of a few large ones.

By adding everything together, the congestion factor for an itinerary is calculated as follows:

Squared Congestion Factor

$$CF_{ri} = \frac{\sum_{a \in A} \sum_{h \in H} \left(V_{ra,h}(i)\right)^2 \sum_{t \in T_h} \delta_{ri,a,t}}{\sum_{a \in A} \sum_{h \in H} \sum_{t \in T_h} \delta_{ri,a,t}}, \quad \forall r \in R, i \in I_r. \tag{6.3}$$

Non-Squared Congestion Factor

$$CF_{ri} = \frac{\sum_{a \in A} \sum_{h \in H} V_{ra,h}(i) \sum_{t \in T_h} \delta_{ri,a,t}}{\sum_{a \in A} \sum_{h \in H} \sum_{t \in T_h} \delta_{ri,a,t}}, \quad \forall r \in R, i \in I_r. \tag{6.4}$$

The $\sum_{t \in T_h}$ is placed after $V_{ra,h}(i)$ because this applies exclusively to $\delta_{ri,a,t}$. This placement follows standard mathematical notation and does not imply any dependence of $V_{ra,h}(i)$ on t.

6.1.3. Algorithm

The algorithm acts dynamically and handles each request sequentially as they come in. Per request the algorithm handles the best approach with the knowledge the algorithm currently has.

Objective for each Request

$$\text{Minimize} \sum_{i \in I_r} x_{ri} Penalty_{ri}, \quad \forall r \in R \tag{6.5}$$

Minimizes the total penalty for selecting itineraries for all requests.

Constraints:

1. **Penalty Function:**

$$Penalty_{ri} = w \left(\frac{\Delta t_{ri}}{120} \right) + (1 - w) \left(CF_{ri} * TF_r \right), \quad \forall r \in R, i \in I_r \tag{6.6}$$

Combines penalties for time deviation and congestion factor, weighted by w, to calculate the penalty for each itinerary.

2. **Itinerary selection for each request:**

Ensure that exactly one option is selected for each request.

$$x_{ri} = \begin{cases} 1, & \text{if itinerary } i \text{ is selected for request } r, \\ 0, & \text{otherwise,} \end{cases} \quad \forall i \in I_r, \, r \in R. \tag{6.7}$$

Defines x_{ri} as a binary variable indicating whether an itinerary is selected.

$$\sum_{i \in I_r} x_{ri} = 1, \quad \forall r \in R \tag{6.8}$$

Ensures exactly one itinerary is selected for each request.

3. **Capacity Constraints for Each Arc a with Departure Time t when handling request r:**

Ensure that the capacity constraints are not violated, including the headway time h_s:

$$\sum_{b=1}^{r} \sum_{i \in I_b} x_{bi} \delta_{bi,a,t} = U_{ra,t'} \quad \forall r \in R, a \in A, t \in T, t' \in [t, t + h_s) \tag{6.9}$$

Updates arc usage by summing allocated itineraries over all previous requests.

$$U_{ra,t} \leq 1 \quad \forall r \in R, a \in A, t \in T \tag{6.10}$$

This constraint accumulates the usage of each arc a with departure time t up to request r, ensuring that once a section is occupied, it cannot be used by subsequent requests beyond its capacity.

4. **Binary Decision Variables:**

$$x_{ri} \in \{0,1\} \quad \forall i \in I_r, r \in R \tag{6.11}$$

Defines x_{ri} as binary variables indicating the selection of itineraries.

6.1.4. Network State Update

After offering an itinerary i to the customer with request r, the network state is updated to reflect the changes in $U_{ra,t}, U_{ra,h}, D_{ra,h}, D_{r,od,t}, D_{ra,t}, D_{rp,t}, V_{ra,h}$ before moving on to the next request $r+1$.

1. Update Number of Trains Departing over Arcs per Hour

Update $U_{ra,h}$ to reflect the new number of trains allocated on arc a during hour h:

$$\sum_{b=1}^{r} \sum_{i \in I_b} x_{bi} \sum_{t \in T_h} \delta_{bi,a,t} = U_{ra,h} \quad \forall a \in A, h \in H, r \in R \tag{6.12}$$

where T_h is the set of times within hour h.

2. Update Forecasted Demand

$$D_{od,t} - \sum_{b=1}^{r} q_b(o,d,t) = D_{r,od,t} \quad \forall o,d \in N, t \in T, r \in R \tag{6.13}$$

Calculates the remaining origin-destination demand $D_{r,od,t}$ for o, d at time t after processing all requests up to r.

From the updated $D_{r,od,t}$ recalculate the demand per section in the network:

Distribute Demand Across Paths: Allocate the OD demand $D_{r,od,t}$ with departure time t among the paths $p \in P_{od}$ based on the calculated probabilities:

$$D_{rp,t} = D_{r,od,t} \times \mathbb{P}_p \quad \forall r \in R, o \in N, d \in N, p \in P_{od}, t \in T \tag{6.14}$$

Update Forecasted Demand per Arc: Decompose each path p into its constituent arcs $a \in A_p$. Remembering from Section 4.2, itinerary i is a combination of p_i and t_i. The set $i \in I$ is built from a combination of the sets $p \in P$ and $t \in T$. So when iterating through all the $i \in I$, would provide the same iteration as doing this separately for $p \in P$ and $t \in T$. This is an important distinction to make so there is no confusion between the departure time t for a path and the departure time t for an arc.

With all trains traveling at a fixed speed v and not stopping at intermediate stations, the arrival time at each node $n \in N_p$ along path p for departure time t can be calculated by:

$$\sum_{i \in I} D_{rp_i,t_i} \delta_{ri,a,t} = D_{ra,t} \quad \forall r \in R, a \in A, t \in T \tag{6.15}$$

Calculates the demand $D_{ra,t}$ for each arc a at time t by summing the contributions of all itineraries i that use the arc, accounting for the demand D_{rp_i,t_i} and whether the itinerary traverses the arc ($\delta_{ri,a,t} = 1$).

Convert Minute-Level Demand to Hour-Level Demand:

$$D_{ra,h} = \sum_{t \in T_h} D_{ra,t} \quad \forall r \in R, a \in A, h \in H \tag{6.16}$$

Aggregates the minute-level demand $D_{ra,t}$ across all minutes t in hour h to compute the hourly demand $D_{ra,h}$ for each arc.

3. Update Congestion Factors

$$V_{ra,h} = \frac{D_{ra,h}}{C_{a,h} - U_{ra,h} + \epsilon} \quad \forall a \in A, h \in H, r \in R \tag{6.17}$$

Recalculates the congestion factor $V_{ra,h}$ for arc a during hour h based on updated demand $D_{ra,h}$, available capacity $C_{a,h} - U_{ra,h}$, and a small constant ϵ to avoid division by zero.

4. Update Time Fraction for Request

$$TF_r = \left(1 - \frac{r}{|R|}\right) \quad \forall r \in R \tag{6.18}$$

Calculates the time fraction TF_r, which decreases linearly as the request index r approaches the total number of requests $|R|$, reducing the weight of congestion over time.

6.1.5. Explanation

The network state update ensures that the allocation of the current request to an itinerary affects the availability and congestion of the network for subsequent requests. By updating $U_{ra,t}$, future itineraries are prevented from using arcs marked as occupied at those times. Updating $D_{a,t}$ and $D_{a,h}$ with $D_{ra,t}$ and $D_{ra,h}$ reflects the decreased demand over the network due to allocated requests, and updating $V_{ra,h}$ allows the model to update the congestion factor, forecasting how congested arcs will be for certain hours. By updating all these parameters, the model can adapt to the changing network conditions throughout the scheduling process.

6.2. Conclusion

This mathematical model provides an algorithm for scheduling requests onto the railway network, forming the basis for the experiments in Chapter 7, where the effects for different values of w are evaluated against ProRail's current system. The model integrates key components such as feasible itinerary generation, congestion factor calculation, and network state updates to dynamically process requests in real-time as they come in.

By introducing an objective function that minimizes both departure time deviations and forecasted congestion and a weight w balancing priority between the two, the model captures the trade-offs inherent in managing limited railway network capacity. By using deterministic demand forecasting, it estimates section-level congestion and provides a predictive framework for allocating requests while maintaining flexibility for future scheduling. By sequentially processing requests and updating the network state after each decision, the model ensures adaptability to real-time changes and lays the foundation for testing scenarios aimed at improving scheduling efficiency and network utilization.

Table 6.1: Notation sets, parameters and variables

Sets	**Description**	**Indexing**
N	Set of nodes (stations) in the network	$n \in N$
A	Set of arcs (links between stations)	$a \in A$
P	Set of predefined paths (sequences of nodes)	$p \in P$
P_{od}	P_{od} is the subset of P containing all paths connecting origin o and destination d	$p \in P_{od}$
T	Set of discrete time points (minutes) within a day	$t \in T$
T_h	Set of discrete time points (minutes) within hour h	$t \in T_h$
H	Set of hours within the day	$h \in H$
r	Set of requests	$r = 1, 2, \ldots, R$
I	Set of itineraries	$i \in I$
I_r	Set of itineraries for request r ($I_r \subset I$)	$i \in I_r$
S	Set of sections (time-dependent arcs)	$s \in S$
Parameters	**Description**	
o_r	Origin station for request r	
d_r	Destination station for request r	
t_r	Preferred departure time for request r	
t_i	Departure time for itinerary i	
p_i	Path used for itinerary i	
v	Fixed speed of trains (e.g. 80 km/h)	
h_s	Minimum headway time for section s	
w	A value between [0,1] for the weight	
$q_r(o,d,t)$	A binary parameter which is 1 if request r contributes to the demand for origin o, destination d, and departure time t; 0 otherwise	
$\mathbb{P}_p$	Probability a request travels over path p	
Δt_{ri}	Time deviation for itinerary i of request r	
CF_{ri}	Congestion factor for itinerary i of request r	
TF_r	A given time fraction for request r	
$Penalty_{ri}$	The penalty for itinerary i for request r	
$D_{od,t}$	Future demand between o and d with departure time t	
$D_{rod,t}$	Left over future demand between o and d with departure time t when scheduling request r	
$D_{a,t}$	Forecasted demand in number of requests preferring arc a with departure time t	
$D_{ra,t}$	Left over forecasted demand in number of requests preferring arc a with departure time t when scheduling request r	
$D_{a,h}$	Forecasted demand in number of requests on arc a during hour h	
$D_{ra,h}$	Left over forecasted demand in number of requests on arc a during hour h when scheduling request r	
$D_{rp,t}$	Left over forecasted demand in number of requests for path p with preferred departure time t when scheduling request r	
$V_{ra,h}$	Congestion factor for arc a during hour h up to request r	
$V_{ra,h}(i)$	Congestion factor for arc a during hour h up to request r, including the allocation of itinerary i to request r	
$U_{ra,t}$	Indicator if arc a with departure time t is occupied when scheduling request r; $U_{ra,t} = 1$ if occupied, 0 otherwise	
$U_{ra,h}$	Value representing number of requests allocated to arc a during hour h up to request r	
$U_{ra,h}(i)$	Value representing number of requests allocated to arc a during hour h up to request r, including the allocation of itinerary i to request r	
$\delta_{ri,a,t}$	1 if itinerary i for request r uses arc a with departure time t, 0 otherwise	
$d_{n,n'}$	Distance between node n and node n'	
ϵ	Small value to avoid division by 0	
Variables	**Description**	
$x_{ri} \in \{0,1\}$	Binary decision variable that equals 1 if itinerary i is offered to request r, 0 otherwise	

7 Case study

This chapter presents a case study to evaluate the proposed dynamic scheduling model for freight trains in the Netherlands. Real-world data is used to simulate the scheduling process, and the performance of the new model is compared with the current scheduling method used by ProRail. Key performance indicators (KPIs) detailed in 1.2 are used to assess the effectiveness of the proposed model.

The primary objective of the case study is to measure the impact on the total time deviation squared after allocating all requests onto the network under various scenarios, including different network capacities, squaring congestion factor values, scaling the congestion penalties over time and altering the request order.

The case study applies the dynamic scheduling model across these varied conditions and for each condition, determines the optimal value of w for the objective function 6.6. The goal is to identify the optimal value of w which minimizes total departure time deviation squared after allocating all requests onto the network.

This chapter is structured as follows: the data gathered for the model is first introduced, along with its role in the algorithm. This is followed by a description of the preprocessing steps, transforming the data into the necessary sets and parameters. Next, the experimental setup is outlined, covering the formulas used for calculating travel times, generating feasible itineraries, and determining penalties. The algorithm then sequentially processes each request, simulating real-time scheduling. Multiple experiments are executed, with results displayed in tables and graphs. The chapter concludes with an analysis of these results and potential applications for optimizing scheduling.

7.1. General description

The case study evaluates the proposed dynamic scheduling model for freight trains in the Netherlands using real-world data. It examines the model's ability to allocate train requests onto a constrained network while balancing customer departure preferences and network congestion. The study simulates scheduling under varying network capacities and parameter configurations to test the robustness and efficiency of the model. Results are analyzed to assess the impact of different weights in the objective function, providing insights into the trade-offs between minimizing total time deviation and mitigating congestion.

7.1.1. Coding

The coding for this project is all done in Python, using Spyder IDE (version 3.11) (Raybaut, 2009).

For the implementation of the model, several libraries were used. The algorithm was carried out using Gurobi, a powerful optimization solver designed for integer programming problems (Gurobi Optimization, 2024). Data manipulation and preprocessing were handled using the Pandas library (pandas development team, 2020). Numerical computations relied on NumPy, which provides efficient array operations and mathematical functions (Harris et al., 2020). Visualizations of the results were done using Matplotlib and its 3D toolkit Axes3D (Hunter, 2007), as well as Plotly for creating interactive graphs (Inc., 2024). Data serialization and storage were performed using Python's built-in pickle module.

7.2. Data description

This section introduces the datasets used in the case study and their role in constructing the scheduling model. The data includes freight train requests, predefined freight paths, and a distance matrix representing the railway network. Each dataset is preprocessed to address missing information and ensure compatibility with the model. These inputs form the foundation for creating the sets, parameters and demand distribution required for the scheduling algorithm.

7.2.1. Requests

The main data used in this case study consists of freight train requests submitted for an average weekday with minimal track maintenance activities (ProRail, 2024b), specifically on *12 March 2024*. There are a total of 953 requests handed in for this day. The requests were collected over a submission horizon spanning multiple months. A small section of the data is pasted in Table 7.1 where the first four requests are shown.

Each request includes the following information:
Order number and **train number**: Used to distinguish between different requests. Each request has a unique combination of order number **and** train number. Multiple requests may share the same order number or train number, so both numbers are taken into account to differentiate between requests.
Date: Preferred departure date.
Follow number: Defines the sequence of stops in the train's route, where the smallest and largest numbers indicate the origin and destination. Intermediate values denote additional stops, either specified by the customer for route constraints or for picking up/dropping off containers. The starting value of the follow number (either 0 or 1) depends on the portal through which the request was submitted.
Service point: The abbreviated names of the stations. The corresponding station names are listed on the website (Spilt, n.d.).
Plan time: Indicates the preferred departure time in seconds from 00:00, with the final station's time indicating preferred arrival. In this thesis only the starting departure time is taken into account.
Offset: A value of 1 indicates that the plan time crosses over into the next day (13-3-2024), otherwise this value is set to 0.

Table 7.1: Raw Train Request Data

Order number	Train number	Date	Follow number	Service point	Plan time	Offset
495740	40065	12-3-2024	1	Whz	48360	0
495740	40065	12-3-2024	2	Kfhn	49200	0
495740	40065	12-3-2024	3	Brmet	51720	0
495740	40065	12-3-2024	4	Zvg	54360	0
495740	40065	12-3-2024	5	Zvg	54960	0
495740	40065	12-3-2024	6	Zvg	55080	0
504604	55561	12-3-2024	0	Wgm	3600	0
504604	55561	12-3-2024	1	Wgm	3600	0
505667	55565	12-3-2024	0	Whz	39631	0
505667	55565	12-3-2024	1	Ps		
505667	55565	12-3-2024	2	Mvtww	45037	0
515226	51026	12-3-2024	0	Vl	81881	0
515226	51026	12-3-2024	1	Lutdsm		

Some irregularities are observed in the data, which occur frequently enough to be treated as standard cases in the model. For example, repetitive intermediate stations (e.g., 'Zvg' listed three times consecutively for a single request), however, only the origin and destination (smallest and biggest number in the follow number column) are used in the model so no action has to be taken to fix this. Requests with identical origin and destination points are discarded. Additionally, only the preferred departure time is used, as arrival times are inconsistently provided. With a fixed uniform speed for all trains, using the departure time is sufficient, as travel time remains constant regardless of the specified arrival time.

7.2.2. Freight paths

A list of the most frequently used freight paths is provided by ProRail. Between the most used origin-destination pairs, ProRail computed (the best) options to connect an OD pair. This saves computation time and gets rid of any cycle paths or paths with big detours. The multiple options of freight paths which can be used between an OD pair are also expected to be similar in travel time. Each path consists of a list of every single station a train would need to pass to reach its destination. In Table 7.2, two different and independent predefined paths are shown. The 'Service Point' column shows the abbreviated station names of the path. The 'Activity' column shows the origin (V) of a station, the stations the path passes (D), and the destination (A) of a station.

Table 7.2: Train Service Points and Activities

Service Point	Activity
Rtd	V
Rhsla	D
Hsrtdt	D
Hsvbw	D
Hsrtdv	D
Hsghtz	D
Hsghtn	D
Hshmdo	D
Hshfdo	D
Hfdm	D
Hfd	D
Shl	A
Shl	V
Asra	D
Asdl	D
Aeg	D
Ass	D
Asdta	D
Sgra	D
Sgbr	D
Asd	A

7.2.3. Distance Matrix

In Table 7.3, a small section of the first ten stations with corresponding distances are shown. The distance matrix shows the distance between every pair of stations in the Netherlands. It represents the shortest possible distance between two stations in kilometers. This data is gathered from the website (Rijdende Treinen, 2022).

Table 7.3: Station Distance Matrix

Station	AC	AH	AHP	AHPR	AHZ	AKL	AKM	ALM	ALMB	ALMM
AC	XXX	82	83	85	90	71	188	32	38	31
AH	82	XXX	1	3	8	77	153	98	104	97
AHP	83	1	XXX	2	9	78	152	99	105	98
AHPR	85	3	2	XXX	11	80	150	101	107	100
AHZ	90	8	9	11	XXX	69	161	106	112	105
AKL	71	77	78	80	69	XXX	211	96	102	95
AKM	188	153	152	150	161	211	XXX	158	152	159
ALM	32	98	99	101	106	96	158	XXX	6	1
ALMB	38	104	105	107	112	102	152	6	XXX	7
ALMM	31	97	98	100	105	95	159	1	7	XXX

The problem with the data is that this is meant mostly for passenger trains and stations. Freight trains may use different stations and paths which are not specified in this matrix. I was not able to gather a distance matrix which contained distances between all the stations in the Netherlands, so to bypass this problem, whenever the distance between two subsequent stations is not known, a default distance is implemented of 5.1 kilometers. This is based on the average distance between two train stations in the Netherlands (Compendium voor de Leefomgeving, 2018).

7.2.4. Station Coordinates

Geographic coordinates for most known stations are gathered from the same website as the distance matrix (Rijdende Treinen, 2022). The coordinates are used only for visualization purposes to plot the stations and are not needed for the actual experiment conducted in this thesis. Also the same problem is present for the list of coordinates, where the coordinates are not known for all the stations used in this research.

7.2.5. Missing Data

Rail Capacity: Due to data limitations, specific capacity data for each arc could not be obtained for this thesis. Rail capacity is influenced by various factors and is challenging to calculate for individual arcs. It is primarily determined by headway time, the safety interval required between two consecutive trains traveling in the same direction from the same station. Headway time varies based on factors such as train speed, station size, number of connecting tracks, and whether tracks are single or bi-directional. For example, the theoretical maximum capacity of the London Underground is 33 trains per hour, while in the Netherlands, eight trains per hour over an arc can already be considered as high (OVNet, 2023). At least a minimum interval of three minutes between consecutive trains on the same track is required (Treinreiziger, 2020), which would result in a maximum of twenty trains per hour. In practice, this interval is almost never achieved. On the betuweroute the amount of trains passing per hour is closer to three trains per hour (RTV Dordrecht, 2024).

In this thesis, a uniform headway time is applied across all sections, simplifying capacity to a constant value for each station or arc. This approach removes natural variations in capacity across routes, such as the higher capacity around the port of Rotterdam compared to less-used sections in the north-east of the Netherlands, but is necessary due to the missing data. Since the exact headway time is unknown, multiple scenarios with different headway times are tested to evaluate the model's performance under varying conditions.

Velocity In this thesis a uniform velocity is used. The velocity which is used is set to an average of 80 km/u. This is based on the websites (ProRail, n.d.) and (Compendium voor de Leefomgeving, 2018).

7.3. Data Preprocessing

The data preprocessing section addresses the data gaps and formatting issues encountered during the preparation of the freight scheduling model. Each preprocessing step uses the data inputs from Section 7.2 to help construct the sets and parameters mentioned in the Chapter 6. This consists of defining the paths, structuring requests, estimate demand distribution, and distance calculation.

7.3.1. Data Loading and Initial Processing

The datasets consist of freight train orders, paths used by infrastructure managers, and distance data, loaded from separate Excel files. Paths are sourced from multiple data files, with each one containing specific routes or segments used for freight transport. These paths, along with order requests and distance data, are imported and initialized to begin constructing the model's parameters.

7.3.2. Node Identification

Unique nodes or stations are derived from the path data to create a foundational list of locations in the network. This list of nodes serves as a reference for constructing paths and allocating orders, ensuring that all nodes included in the dataset are accounted for and identifiable.

7.3.3. Path Extraction

The path processing function transforms sequences of ”Dienstregelpunt” (service points) into unique paths with a clear starting station ('V'), intermediate stations ('D'), and end stations ('A') as can be seen in Table 7.2. The function iterates through each row of the path data to construct paths by: Starting a path when an activity marker is 'V', appending intermediate nodes, ensuring no consecutive duplicates, and ending the path when 'A' is encountered. Then convert the resulting list into a tuple for uniqueness. After processing each data file, the unique paths are combined across datasets to avoid redundancy. This results in a set of freight paths, denoted as P, which serves as the foundation for route options to fulfill requests.

7.3.4. Structure Requests

From the request dataset (see Table 7.1), all data is gathered, stored and labeled in a dictionary. The requests are grouped by ”ordernumber” and ”trainnumber”. Each request consists of multiple stations, of which there is at least an origin and a destination, and a preferred departure time (noted in seconds from midnight). Each processed request is then stored in a dictionary list of requests: R, which can then be used for the demand forecast calculation, testing the path allocation and later on to conduct the actual experiment with.

7.3.5. Demand Calculation

Demand is calculated by counting each OD pair's frequency from all the requests R and also storing the time: denoted as D_od_t. This dataset maps the expected number of requests for each OD pair across different time intervals. This list will be used to check which paths can be used to connect the OD pairs, and for the calculation of the demand forecast over the whole network.

7.3.6. Path Extension and Creation

From the initial data analysis, it became clear that a significant number of requests (approximately 64%) could not be allocated to the predefined freight paths stored in P. This means there is a substantial lack of paths to satisfy origin-destination (OD) pairs. From meetings with employees from CGI and ProRail, I was already warned for this. Also in the real life scheduling process, approximately 40% of the requests can be allocated directly to a path. The other 60% needs to be manually created by employees.

To increase the number paths which can fulfill requests, three steps are taken:

1. Subpath Extraction: For OD pairs lacking direct paths, subpaths are created by extracting segments from existing paths where both origin and destination nodes appear. If a suitable segment exists, it is included as a path for the OD pair. OD pairs which can still not be matched to a (sub)path are flagged for additional processing.
2. Path Reversal: For the flagged OD pairs, paths are reversed to assess suitability in the opposite direction. Reversing paths provides additional path options by allowing request to travel in reverse order. This is not possible in real life due to some rail sections being one-directional. For this study, reversing paths is (somewhat) justified as the primary aim is to test an algorithm to examine total departure time difference, rather than trying to perfectly simulate a real world scenario.
3. Path Combination: For OD pairs that still can not be allocated to a path after subpath extraction and reversal, the model combines paths to create extended, non-cyclic routes between endpoints.

This method involves merging paths that share either the same start or end points. Although some combined paths may not be optimal and could involve detours, they provide a necessary alternative to allow the train to reach its destination. In this context, ensuring connectivity takes precedence over routing efficiency, as the primary goal is to fulfill the request even if the resulting path is longer than ideal. If multiple paths are possible to fulfil an OD pair, only the shortest feasible path is used.

The result after executing these steps is an expanded path set P that can fulfill more OD pairs, allowing for a more comprehensive allocation of paths to requests.

After generating the new paths, the model assesses each OD pair to determine matching status. OD pairs which can still not be fulfilled to one of the available paths are flagged and removed from the demand and request datasets, reducing computational complexity by retaining only feasible requests for allocation.

The path expansion steps significantly increase the number of matched requests. Of the original 953 requests, 767 can now be allocated to one or more paths. The remaining 186 requests, for which no path is available, are removed from the request set R.

7.3.7. Arc Creation

Unique arcs, representing individual segments between sequential nodes, are identified by iterating through the final path set P. Each arc is defined as a direct connection between two consecutive nodes within a path. These arcs are stored as unique entries in A.

7.3.8. Distance Matrix Filtering

Using the provided distance matrix, known distances between stations are stored in a filtered dictionary. For arcs without distance data, a default value of 5.1 km is assigned to maintain continuity.

7.3.9. Demand Distribution by Section

The distribution of demand across network sections is handled by applying the Commonality Factor Logit (C-Logit) model, as explained in Subsection 5.1.1. This method accounts for path overlap by assigning probabilities to each path based on utilities derived from path attributes, in this case specifically the total overlap distance between paths. For all paths which can connect an origin-destination (OD) pair, a utility score that reflects their degree of overlap with the other paths is calculated. Overlapping parts are identified using the algorithm in Subsection 5.1.1.4, quantifying overlap by measuring the distance between shared arcs. This approach ensures that the demand is distributed fairly over the available paths in the network, by penalizing paths with high overlap.

After calculating the path probabilities, demand for each OD pair is distributed across the feasible paths based on these probabilities. For the forecasted demand assigned to a single path, demand is allocated in more detail when zooming into the sections in the path. The sections are also time dependent, so the demand allocated to a path should also account for the travel time as trains may span multiple hours in transit. This is simplified in hour-by-hour occupancy per section. (This is explained in more detail in 5.1.1.) Demand for each section along a path is accumulated based on the hour during which a train is expected to occupy that section. After executing this process for all OD pairs, the total expected demand for each section in the network for each hour is known.

This demand distribution model considers both route overlap and travel time, ensuring that demand is distributed over paths/sections in a way that is simple, yet reliable. In Table 7.4a the top 30 forecasted busiest arcs are shown. With the busiest arc being (Whzan, Brdv) with a forecasted demand of 86 trains for one day. When using the website of Spilt (n.d.), the arc (Whzan, Brdv) is translated to the arc going from station 'Rotterdam Waalhaven Zuid Aansluiting' to 'Barendrecht Vork'. When looking at the railway map provided by ProRail (ProRail, 2025) it becomes clear why this is the forecasted busiest arc for freight trains on a day: all trains leaving the port of Rotterdam, travel over this arc.

7.3.10. Data Storage

All preprocessed data, including nodes, arcs, paths, requests, demand distribution, commonality factors and distance per arc is stored in a dictionary and saved as a pickle file. This final storage step preserves the processed data for efficient retrieval and model execution, ensuring that the algorithm can be run on a fully preprocessed dataset.

7.3.11. Computation time

Computation time is approximately 35 seconds.

7.4. Model Implementation

The following section shows how the algorithm used for freight train scheduling is implemented using python, linking each coding step to the corresponding mathematical or conceptual model. Choices or simplifications made during the coding are explained or justified in this Section.

7.4.1. Data loading and preprocessing

The algorithm starts by loading the preprocessed data stored in a pickle file from Section 7.3. The data includes the sets of nodes (N), arcs (A), predefined paths (P), and freight train requests (R). Time intervals are represented in hours (H) and minutes (T). Demand data includes $D_{od,t}$ for origin-destination pairs and departure time in minutes and $D_{a,h}$ for the demand per arc per hour. Also the $C-logit$ data is imported so during the scheduling process the forecasted demand can be recalculated easily. Additionally, a distance matrix (dm) and precomputed travel times for paths ($path\_travel\_times$) are loaded.

A constant train speed ($v = 80km/h$) is assumed for simplicity, ensuring uniform travel times across all paths. This assumption aligns with the mathematical model, where operational variations in speed are disregarded to focus on improving the departure time deviation during scheduling. The initialization also includes a minimum follow up time (h_s) which is used to calculate capacity constraints for sections in the railway network. The headway time will be altered for multiple experiments to see how the algorithm performs under different capacity constraints in the network.

The maximum network capacity for each arc per hour $C_{a,h}$ and the congestion factor for each arc per hour $V_{a,h}$ are calculated during initialization to represent the availability and anticipated congestion levels for each arc per hour. These values are based on the preprocessed data and provide inputs for scheduling decisions. These values are calculated in the same way as presented in subsections 5.1.2 and 5.1.3.

7.4.2. Function definitions

For the algorithm, three functions are coded which can be called for when needed.

1. **Travel time calculation:** The function $compute\_travel\_times$ calculates travel times between stations along a path using the distance matrix dm and the speed v. Travel times are expressed in minutes, ensuring consistency with the scheduling time units.
2. **Feasible itinerary generation:** The function $generate\_feasible\_itineraries$ identifies and returns all feasible itineraries for a given request r, considering paths that match the request's origin and destination and checking their capacity availability in $U_{ra,t}$. Feasibility is evaluated for departure times within hourly increments forward and backward from the preferred departure time, making use of the fact that penalties are the same across the hour. Each itinerary is validated by iterating through its sections and ensuring sufficient capacity is available during the train's traversal, accounting for headway times and the velocity of the train. When a feasible itinerary is found within an hour, the algorithm skips to the next hour to save computation time. The function efficiently handles forward and backward searches by iterating through all time increments until the options are explored or the time range is exhausted.
3. **Congestion factor calculation:** The function $calculate\_congestion\_factor$ computes the congestion factor for an itinerary by using the penalties for all sections in the itinerary, considering the departure time and train velocity to determine the specific hours for when each section is traversed. The congestion factor for each arc is retrieved from $V_{ra,h}$, with the arc-hour pair as the key and the request r referring to the request in question. The function supports two methods for calculating the congestion factor: one computes the average congestion by summing all the congestion values for all sections in the itinerary and dividing by the total number of sections. The

other method squares the congestion factor before summing them, creating a bigger difference between large and small congestion factors. The division by the total number of sections remains the same in both methods. These approaches are based on the equations presented in Subsection 6.1.2, and experiments in 7.5.3 compare their impact on algorithm performance. The output of the function is a single value for the congestion factor of the given itinerary.

7.4.3. Request processing algorithm

Each request is processed sequentially, updating the network state ($U_{ra,t}$, $U_{ra,h}$, $D_{rod,t}$, $D_{ra,h}$ & $V_{ra,h}$) after each allocation. This approach mimics real-time scheduling and ensures that future decisions account for previously allocated requests. The process of handling a request follows the same steps as seen in Figure 4.1.

Once a request r gets handed in, the first step is generating all feasible itineraries for the request using the function $generate\_feasible\_itineraries$. For each feasible itinerary its time deviation from the preferred departure time and the congestion factor corresponding to the capacity, network utilization and forecasted demand are calculated. All the feasible itineraries with their time deviation and congestion factor are used as an input for the objective function.

The algorithm uses the binary decision variable (x_{ir}) to indicate whether an itinerary is selected for a request (r). The objective function minimizes the penalty resulting from a weighted sum of normalized time deviation and the congestion factor. A weight parameter (w) determines the trade-off between these two components. The objective function is expressed as:

$$\text{Minimize} \quad \sum x_{ir} Penalty_{ir}, \quad \forall r \in R$$

For the experiments, two variations of the congestion factor are tested. In one variation, the congestion factor remains constant throughout the scheduling horizon. In the other, the congestion factor penalty decreases over time, reducing its impact for later requests. The penalty for each itinerary, where the congestion factor stays the same, is calculated as:

$$Penalty_{ri} = w\left(\frac{\Delta t_{ri}}{120}\right) + (1-w)\left(CF_{ri}\right), \quad \forall r \in R, i \in I_r$$

For scenarios where the congestion factor decreases over time, the model incorporates a time fraction (TF_r) to adjust the relative importance of congestion penalties for later requests. The weight (w) still balances the trade-off between time deviation and network utilization. This variant modifies the penalty calculation as follows:

$$Penalty_{ri} = w\left(\frac{\Delta t_{ri}}{120}\right) + (1-w)\left(CF_{ri} * TF_r\right), \quad \forall r \in R, i \in I_r$$

The model includes constraints ensuring that exactly one itinerary is selected per request. Capacity constraints are enforced by checking the availability of sections at all required times, including headway periods. If a section turns out to be unavailable, the corresponding itinerary is excluded from the solution.

Once the model is solved, the selected itinerary is stored, and the network state is updated. The utilization of sections ($U_{ra,t}$) are marked as either 'active' or 'headway' based on the train's occupancy and the headway period. When they are marked with either, these sections, at the specific times are unavailable for future trains to depart over. Demand data ($D_{rod,t}$) is updated by reducing the demand for the scheduled origin-destination pair. Congestion factor values ($V_{ra,h}$) are recalculated for all sections in the network to reflect the updated network state.

7.4.4. Computation time

Computation time between 15 and 24 seconds. Lower network capacity results in the algorithm finding it more difficult to allocate requests to itineraries resulting in larger computation times.

7.5. Experiments

This section evaluates the performance of the proposed mathematical model through a series of experiments. The experiments are designed to analyze the scheduling outcomes, sensitivity to parameter changes, and overall effectiveness of the model in reducing substantial departure time deviations.

The section begins by testing the model under controlled conditions to visualize how requests are allocated throughout the scheduling horizon and to examine the impact of different weights (w) on the scheduling results (Subsection 7.5.1). Insights from these tests are then used to conduct a detailed sensitivity analysis (Subsection 7.5.2), exploring the influence of various parameters, such as network capacity, congestion factor adjustments, and request order. Through the sensitivity analysis the optimal value for w is calculated for individual independent cases.

Subsequent experiments expand on this analysis by repeating the tests with randomized request orders to evaluate the consistency of the results (Subsection 7.5.3). The findings are compared against a base case scenario to identify optimal parameter settings and to assess the robustness of the model under different conditions. Through these experiments, the following question will be answered: is there a weight w, where the outcome will, on average, outperform the base case scenario?

7.5.1. Testing and Visualizing the Scheduling Model

This subsection evaluates the model's performance under controlled conditions to understand its behavior across various scenarios. The tests use a fixed request order unless explicitly stated otherwise to show how different parameters in the model change the scheduling process for the same request list and order.

The first analysis examines the departure time deviation for scheduled requests throughout the scheduling horizon at different weight (w) values. The results demonstrate for the base case scenario ($w = 1$) how network capacity diminishes as scheduling progresses, leading to increasingly larger deviations for later requests. An alternative scenario is also explored, prioritizing congestion factor minimization ($w = 0.1$), which shows that now also earlier requests are occasionally being offered itineraries with departure time deviations.

For both scenario's the results are analyzed and discussed. By comparing the results, the sensitivity of the model to changes in w are highlighted, offering insights into the trade-offs between minimizing time deviation and congestion factor minimization.

Departure Time Deviation Per Request

The base case scenario is first considered, with network capacity set to 5 trains per hour ($w = 1$). The congestion factor does not have an impact in this scenario.

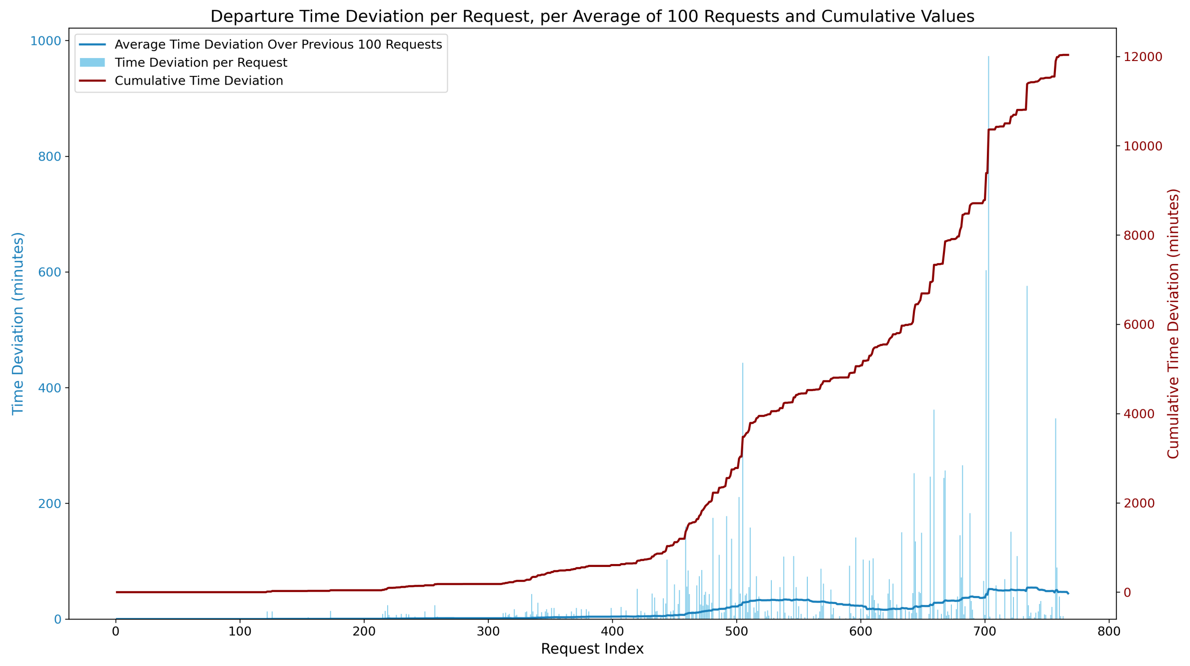


Figure 7.1: Time deviation per request, average time deviation per 100 requests and the cumulative time deviation for $w = 1$

In Figure 7.1 the cumulative time deviation is 12,035 minutes. Early requests are allocated with minimal departure time deviations, but deviations increase significantly near the end of the horizon due to reduced network capacity. Notably, request 704 experiences a deviation of 1003 minutes, and a total of 13 requests exceed a deviation of 180 minutes.

An alternative scenario with $w = 0.1$ is then analyzed under identical conditions: section capacity of 5 trains per hour, squared congestion factor, and congestion factor scaling down per request.

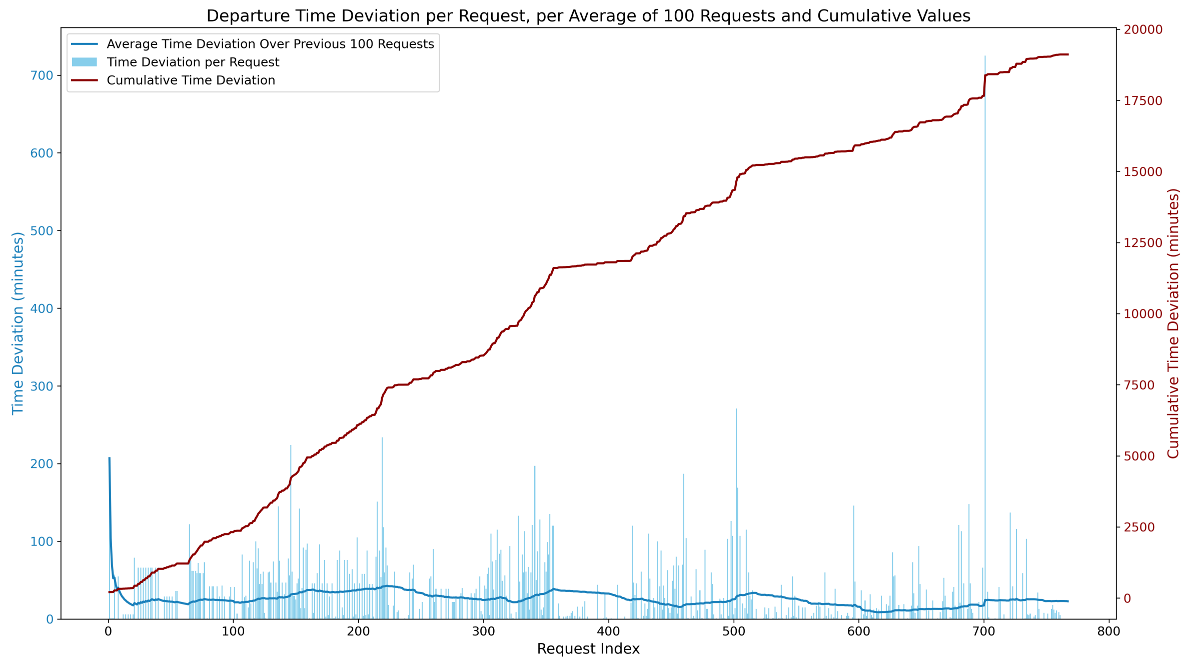


Figure 7.2: Time deviation per request, average time deviation per 100 requests and the cumulative time deviation for $w = 0.1$

In Figure 7.2, the cumulative time deviation increases to 19,114 minutes. Offering alternatives occurs consistently and more often throughout the horizon. In this example, tactically offering more alternatives with small deviations, leads to fewer extreme deviations. There are now a total of 7 requests with a departure time deviation bigger than 180 minutes, the largest being 725 minutes for request 702, but there are significantly more smaller deviations. This behavior highlights the trade-off between either reducing extreme departure time deviations or increasing the cumulative time deviation.

When comparing the two scenario's of Figure 7.1 and Figure 7.2, the scheduling model behaves as expected. The comparison of these two models raises the question of whether an intermediate w value could balance these trade-offs, minimizing large deviations while maintaining an acceptable cumulative departure time deviation.

Forecasted Demand versus Allocated Requests
Table 7.4a presents the forecasted demand per arc. After allocating all requests to itineraries, the total allocated requests per arc can be analyzed to assess the algorithm's impact. Comparing Tables 7.4a, 7.4b, and 7.4c highlights the effects of different w values.

Some arcs serve as critical connections or passages between origins and destinations, meaning that this is the only connection and no alternative arcs are possible. This is shown for some of the cases where the forecasted demand is equal to the number of allocated requests for an arc. While this equality does not necessarily imply the arc is the only passage, but when looking at the railway map of the Netherlands (ProRail, 2025) and seeing it is the only passage, the equality can be justified. For instance, the busiest arc, (Whzan, Brdv), is the only connection between the port of Rotterdam and the rest of the network, and its total usage matches the forecasted demand, as expected.

Time Space Graph
To visualize the allocated requests, time-space graphs are plotted in Appendix C. Plotting all requests for a day quickly becomes overwhelming and difficult to interpret. Therefore, the focus is placed on the most used arc, Whzan-Brdv, which accounts for 86 allocated itineraries. Figures C.1, C.2, and C.3 show time-space graphs for this arc under three different values of w. These graphs illustrate how closely packed the itineraries need to be to fit everything within the day's schedule. The headway time constraints are also visualized by the 12 minutes which need to at least be in between two plotted arcs. While primarily used for visualization, the graphs are not particularly insightful for drawing detailed conclusions. However, a comparison of the graphs for $w = 1$ and $w = 0$ reveals that, with $w = 0$, more itineraries are scheduled at night. This shift occurs because congestion is lowest during nighttime, and the algorithm prioritizes less desirable time slots. Understandably, most requests are submitted for daytime travel.

In Figures C.4 and C.5, all itineraries are displayed in a 3D graph. This approach was chosen because a 2D representation would be overly cluttered. However, the 3D visualization does not fully resolve this issue, as the sheer volume of freight trains scheduled for a single day makes it challenging to interpret. From a top-down perspective, the stations in these graphs mirror the geography of the Netherlands, with notable nodes such as Maastricht (bottom right), Brussels (bottom left), Amsterdam (top left), and Assen (top right) providing geographical context.

Load Distribution Forecasted Demand
To estimate network activity or the load throughout the day, the total number of trains is calculated per time step. Initially, this was done per minute, but the resulting graph was too chaotic to interpret. To address this, the average number of forecasted trains in the network for every five minutes is calculated and visualized. The resulting graph represents the load distribution over the network in an ideal scenario where all customers have zero departure time deviation.

The graph is generated using the demand data for origin-destination pairs requesting specific departure times. Requests are distributed across the predefined paths using the commonality factor explained in Subsection 5.1.1. For each path, the train's velocity is used to calculate its presence in the network over time. Combining this with the percentage of demand distribution per path allows for estimating the time intervals during which trains occupy the network. By repeating this process for all requests, the total train presence in the network can be calculated per minute, which is then averaged over five-minute intervals. The resulting visualization is shown in Figure 7.3.

Table 7.4: Comparison of forecasted demand per arc and total allocated requests per arc for a day

(a) Total forecasted demand per arc for a day

Arc	Forecasted demand
('Whzan', 'Brdv')	86.0
('Kfhan', 'Kfhn')	83.0
('Brech', 'Bropo')	78.6
('Brmet', 'Brech')	78.6
('Bropo', 'Brvalw')	78.6
('Kfhn', 'Kfhz')	77.0
('Brvalw', 'Brvalo')	76.6
('Brdv', 'Whzan')	76.0
('Brdv', 'Kfhan')	75.5
('Zvbtwa', 'Zvo')	75.0
('Zvo', 'Zvg')	75.0
('Brdvno', 'Zvbtwa')	74.6
('Brvalo', 'Brdvno')	74.6
('Kfhn', 'Kfhan')	69.0
('Brech', 'Brmet')	67.6
('Bropo', 'Brech')	67.6
('Brvalo', 'Brvalw')	67.6
('Brvalw', 'Bropo')	67.6
('Zvg', 'Zvo')	67.0
('Zvo', 'Zvbtwa')	67.0
('Brdvno', 'Brvalo')	66.6
('Zvbtwa', 'Brdvno')	66.6
('Kfhz', 'Kfhn')	63.0
('Brgnd', 'Brgro')	62.0
('Brgro', 'Brmet')	62.0
('Brppd', 'Brgnd')	62.0
('Kfhz', 'Brppd')	62.0
('Kfhan', 'Brdv')	60.0
('Sgbr', 'Asd')	54.0
('Ps', 'Rscwa')	53.0

(b) Total amount of requests per arc for a day when $w = 0.1$

Arc	Total requests
('Whzan', 'Brdv')	86
('Kfhan', 'Kfhn')	83
('Brech', 'Bropo')	78
('Brmet', 'Brech')	78
('Bropo', 'Brvalw')	78
('Kfhn', 'Kfhz')	77
('Brdv', 'Whzan')	76
('Brvalw', 'Brvalo')	76
('Zvbtwa', 'Zvo')	75
('Zvo', 'Zvg')	75
('Brdvno', 'Zvbtwa')	74
('Brvalo', 'Brdvno')	74
('Brdv', 'Kfhan')	70
('Kfhn', 'Kfhan')	69
('Brech', 'Brmet')	67
('Bropo', 'Brech')	67
('Brvalo', 'Brvalw')	67
('Brvalw', 'Bropo')	67
('Zvg', 'Zvo')	67
('Zvo', 'Zvbtwa')	67
('Brdvno', 'Brvalo')	66
('Zvbtwa', 'Brdvno')	66
('Kfhz', 'Kfhn')	63
('Brgnd', 'Brgro')	62
('Brgro', 'Brmet')	62
('Brppd', 'Brgnd')	62
('Kfhz', 'Brppd')	62
('Kfhan', 'Brdv')	56
('Sgbr', 'Asd')	54
('Ps', 'Rscwa')	53

(c) Total amount of requests per arc for a day when $w = 1$

Arc	Total requests
('Whzan', 'Brdv')	86
('Kfhan', 'Kfhn')	83
('Brech', 'Bropo')	79
('Brmet', 'Brech')	79
('Bropo', 'Brvalw')	79
('Brdv', 'Kfhan')	78
('Brvalw', 'Brvalo')	77
('Kfhn', 'Kfhz')	77
('Brdv', 'Whzan')	76
('Brdvno', 'Zvbtwa')	75
('Brvalo', 'Brdvno')	75
('Zvbtwa', 'Zvo')	75
('Zvo', 'Zvg')	75
('Kfhn', 'Kfhan')	69
('Kfhan', 'Brdv')	68
('Brech', 'Brmet')	67
('Bropo', 'Brech')	67
('Brvalo', 'Brvalw')	67
('Brvalw', 'Bropo')	67
('Zvg', 'Zvo')	67
('Zvo', 'Zvbtwa')	67
('Brdvno', 'Brvalo')	66
('Zvbtwa', 'Brdvno')	66
('Kfhz', 'Kfhn')	63
('Brgnd', 'Brgro')	62
('Brgro', 'Brmet')	62
('Brppd', 'Brgnd')	62
('Kfhz', 'Brppd')	62
('Sgbr', 'Asd')	54
('Ps', 'Rscwa')	53

This graph is particularly useful for comparing how the network load is distributed under different scenarios in the sensitivity analysis. However, it is important to note that the peaks in the graph do not necessarily indicate congestion. These peaks represent the total number of trains present across the entire network. If the requests use different paths and arcs, the demand can be met without issue. Conversely, a valley in the graph, such as 15 trains, could represent congestion if all requests follow the same path. Despite these nuances, the graph provides a helpful estimate of expected network activity.

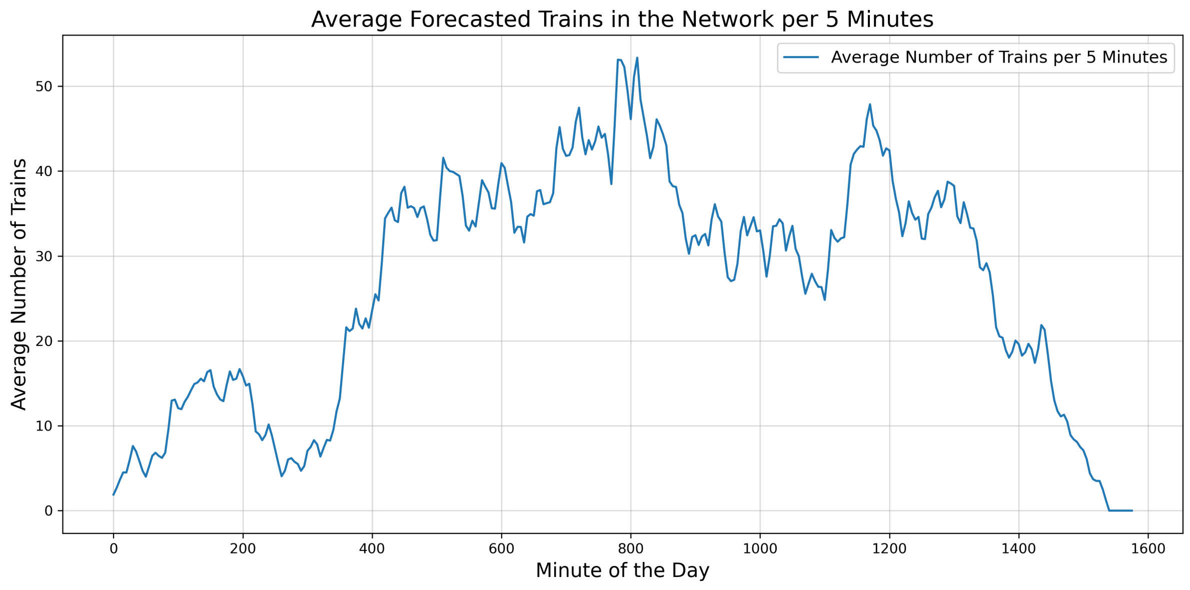


Figure 7.3: Average forecasted trains per 5 minutes

7.5.2. Sensitivity Analysis

This section evaluates the impact of varying the weight parameter w on the scheduling model's performance. By systematically testing w values in the range [0.1, 1.0] with a discrete step size of 0.01, the analysis identifies the optimal weight for minimizing total (cumulative) time deviation and total (cumulative) squared time deviation. The results provide insights into how w influences the trade-off between minimizing overall delays and avoiding extreme outliers in departure time deviation. Additionally, the analysis explores patterns and trends, laying the groundwork for understanding the sensitivity of the model under different scenarios.

For the sensitivity analysis the KPI's presented in Table 1.2 are used as a baseline to try and quantify some of the improvements compared to the baseline. Only the average cumulative departure time deviation and the average cumulative departure time deviation squared are not measured here, but measured in Section 7.5.3.

Only the range of w is tested between 0.1 and 1 due to the fact that implementing a weight value smaller than 0.1 focuses too much on minimizing the congestion factor resulting in extreme time deviations throughout the whole scheduling horizon. This makes the visualization of graphs more difficult and the results are not useful for this thesis.

First Experiment

The first sensitivity analysis is conducted for the following scenario: Capacity = 5 trains per hour, squared congestion factor, congestion factor scaling down per request. The results are plotted in a graph where the weight is on the x-axis and the total (cumulative) time deviation is on the y-axis. The results of the same experiment are also shown in a separate graph where the total time deviation squared is plotted against the weight.

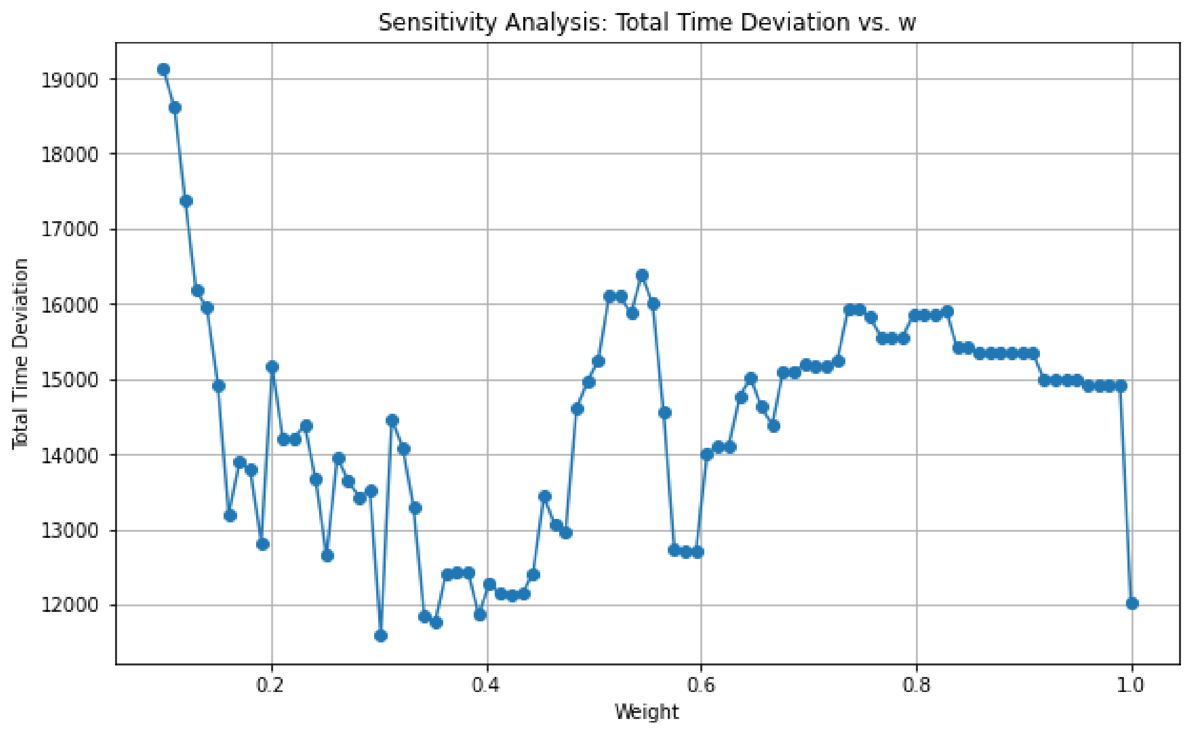


Figure 7.4: Total time deviation plotted per weight

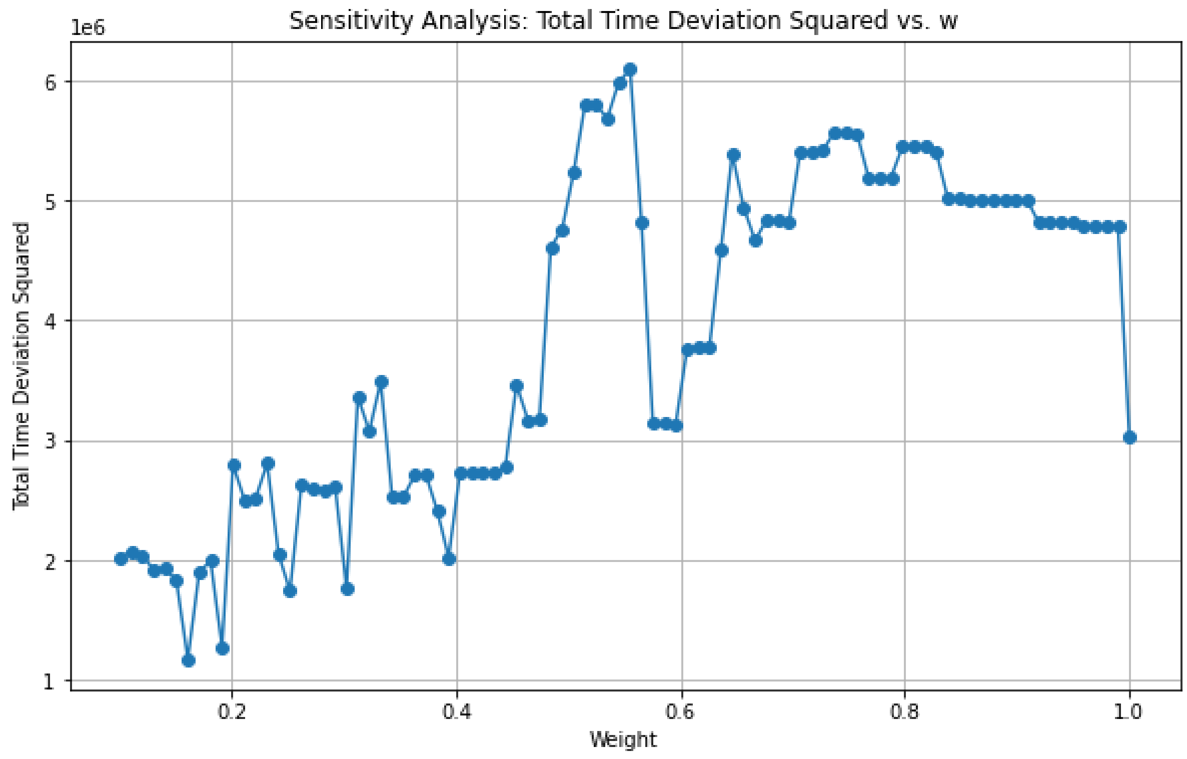


Figure 7.5: Total time deviation squared plotted per weight

Table 7.5: Results for First Experiment (Capacity = 5 Trains/Hour)

Weight	Total Time Deviation (min)	Total Squared Deviation (min)	Number of Requests > 180	Observations
1.00	12,035	3,024,263	13	Base case
0.30	11,593	1,766,177	12	Best total time deviation
0.16	13,211	1,169,785	8	Best squared time deviation

Table 7.6: Results Average and Maximum Time Deviation

Weight	Average Time Deviation (min)	Maximum Time Deviation (min)
1.00	15.78	1003
0.30	14.92	533
0.16	17.22	284

By analyzing the results in Figure 7.4, $w = 0.30$ gives the best result in this scenario with a total departure time deviation of 11,593 minutes which is slightly better than the base case scenario of $w = 1$ which results in a total of 12,035 minutes.

By analyzing the results in Figure 7.5, the best result for the total time deviation squared is for $w = 0.16$ with a value of 1,169,785 minutes which is a lot better compared to the total value of 3,024,263 minutes for the base case when $w = 1$.

The key results for $w = 1$, $w = 0.30$ and $w = 0.16$ are shown in Table 7.5 and 7.6. These results highlight the algorithm's significant impact on extreme departure time deviations. For example, the number of requests exceeding 180 minutes is reduced from 13 in the base case ($w = 1$) to 8 at $w = 0.16$, and the maximum departure time deviation decreases from 1003 minutes to 284 minutes. This demonstrates the algorithm's ability to minimize extreme delays effectively.

However, these improvements come with trade-offs. At $w = 0.16$, while the squared total time deviation is minimized (1,169,785 minutes), the total time deviation increases to 13,211 minutes compared to 12,035 minutes in the base case. Similarly, the average time deviation rises to 17.22 minutes from 15.78 minutes. These results illustrate how the algorithm prioritizes reducing extreme deviations at the expense of slightly higher overall and average deviations.

For $w = 0.30$, the total time deviation is the lowest at 11,593 minutes, achieving a balance between reducing extreme delays and maintaining lower total and average deviations. This demonstrates the algorithm's versatility in adjusting outcomes based on the weight parameter, w, allowing for scenario-specific optimization.

While the total time deviation increases significantly as w approaches 0.1, the squared total time deviation remains relatively low. This presents a trade-off for ProRail: either prioritizing the reduction of extreme time deviations at the expense of higher overall deviations or accommodating more preferred departure times while allowing a few extreme deviations. This correct answer to this trade-off is debatable and is beyond the scope of this thesis.

To visualize how the scheduling changes when inserting the 'best' weight values resulting from the sensitivity analysis. The same scheduling graphs are computed as in Subsection 7.5.1.

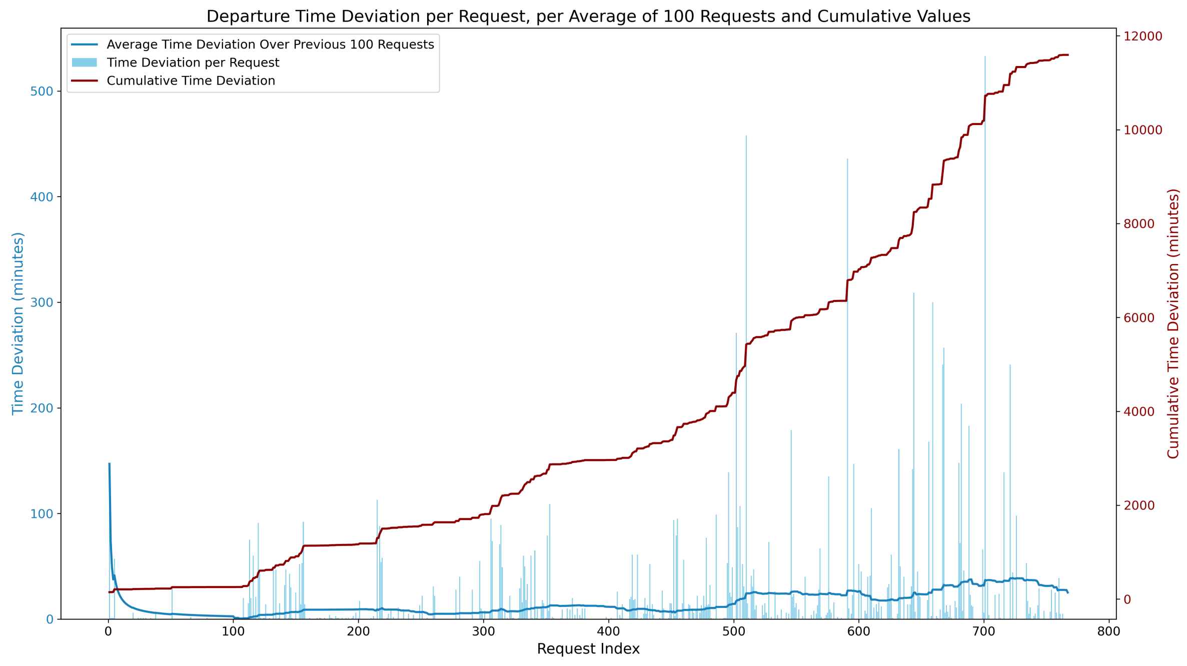


Figure 7.6: Time deviation per request, average time deviation per 100 requests and the cumulative time deviation for $w = 0.30$

When comparing Figure 7.6 to Figure 7.1 (the base case), one big noticeable difference is the scale on the axis for the total time deviation is halved, showing how the number of extreme time deviations is decreased. This graphs shows the optimal result for keeping the total (cumulative) time deviation in a similar range, while also decreasing the extreme time deviations. The total amount of requests above 180 minutes is 12, which is still a similar amount to the base case, but the value of these extreme deviations is decreased.

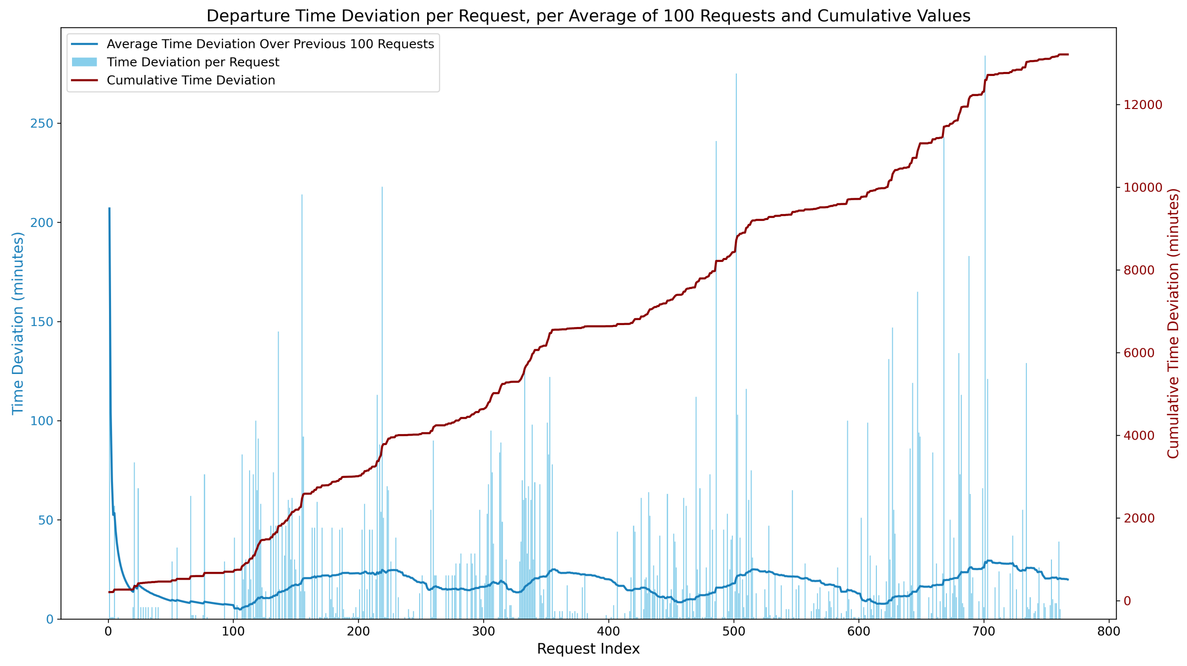


Figure 7.7: Time deviation per request, average time deviation per 100 requests and the cumulative time deviation for $w = 0.16$

When comparing Figure 7.7 to Figure 7.1 (the base case), the most noticeable difference is how the scale on the axis is even smaller compared to the results from Figure 7.2. There are only two requests going over the 250 minute mark. The total cumulative time deviation is slightly larger with a value of 13,211. The total amount of requests with a departure time above 180 minutes is 8.

Appendix D presents the allocation of the first 30 requests in two scenarios: the base case with $w = 1$ and the scenario with the lowest cumulative squared time deviation ($w = 0.16$). By comparing these tables, differences in the algorithm's behavior are noticeable.

In the base case ($w = 1$), all 30 requests are allocated to their preferred departure times, resulting in zero departure time deviation. This is due to the network starting empty and still having enough capacity to distribute everything. In contrast, the $w = 0.16$ scenario actively offers alternatives with departure time deviations to prevent forecasted congestion and improve overall outcomes.

When $w = 0.16$ the algorithm's preference for minor adjustments becomes very active, such as shifting departure times by one minute. For instance, an initial small congestion factor of 0.04, is adjusted by a departure time deviation of one minute to transition the itinerary to a new hour slot where the congestion factor is equal to zero. One might argue that this is unnecessary due to the initial value for the congestion factor was already extremely low. The gain in congestion reduction is minimal, which illustrates the model's strict optimization priorities, which might be too excessive in this scenario.

Another important observation is seen in the handling of the first request. With $w = 1$, the request is allocated to an itinerary with a high congestion penalty of 2.742 due to heavily forecasted congested arcs. In contrast, at $w = 0.16$, an alternative itinerary with a 207-minute departure time deviation and a significantly reduced congestion factor of 0.019 is selected. Table 7.7 shows all feasible itineraries for this request, their corresponding time deviations, congestion factors, and the resulting values from the penalty.

The algorithm selects the itinerary that minimizes the penalty:

$$\text{Penalty} = \frac{w * \text{departure time deviation}}{120} + (1 - w) * \text{congestion factor}$$

In this scenario, the weight $w = 0.16$ places greater emphasis on reducing the congestion factor compared to the time deviation. Among the 24 feasible itineraries (one for each hour along the single path connecting the origin and destination), the itinerary with the lowest penalty is selected, which has a penalty of: 0.292. This occurs because the significant reduction in the congestion penalty outweighs the moderate increase in time deviation.

For example, the itinerary with no time deviation (0 minutes) receives a congestion factor of 2.742, resulting in a total penalty of 2.303. Alternatively, the selected itinerary with a time deviation of 207 minutes and a congestion factor of 0.019 results in the lowest overall penalty of 0.292. This outcome demonstrates how the weighted objective function drives the algorithm's decision, prioritizing a balance between time deviation and congestion factor penalty based on the given weight w.

Table 7.7: All feasible itineraries of first request

Itinerary	Time Deviation	Congestion Factor Penalty	Results Objective Function
1	0	2.742	2.303
2	27	1.767	1.520
3	34	1.299	1.136
4	87	0.508	0.543
5	94	0.550	0.587
6	147	0.228	0.388
7	154	0.543	0.661
8	**207**	**0.019**	**0.292**
9	214	0.454	0.667
10	267	0.215	0.537
11	274	0.044	0.402
12	327	0.060	0.486
13	334	0.545	0.903
14	387	1.198	1.522
15	394	1.246	1.572
16	447	0.906	1.357
17	454	2.087	2.358
18	507	0.329	0.952
19	514	0.868	1.414
20	567	0.078	0.822
21	574	0.208	0.940
22	627	0.009	0.844
23	687	0.539	1.369
24	747	0.009	1.004

Comparing the actual load distribution on the network to the forecasted load distribution, as discussed in Subsection 7.5.1.4, reveals key insights. Figure 7.8 shows the forecasted versus actual load distribution for $w = 1$, representing the base case or first-come, first-served approach. In this scenario, departure times are prioritized. In the graph instances are seen where these priorities cannot be met, leading to redistributions to other time steps. A clear example is seen at the peak around minute 800, the demand cannot be fully satisfied due to insufficient network capacity. The actual number of trains at minute 800 in the network (red line) is lower than the forecasted amount. Between minutes 600 and 700, a forecasted valley indicates available capacity, suggesting that the algorithm shifts requests from minute 800 to this interval to respect capacity constraints, which is seen be the red line showing more allocated requests at this point. The same switch in the redistribution of forecasted capacity versus actual capacity can be seen in between the minutes 1050 and 1200.

When examining the load distribution for $w = 0.16$ in Figure 7.9, the algorithm's behavior becomes more apparent. The algorithm actively avoids forecasted congestion, steering requests towards less busy sections. The workings of the algorithm can be seen in its extreme form when the values for w are reduced even more, as seen in Figure E.1 for $w = 0$. In Figure 7.9, the actual allocation of requests near the 800-minute mark is further reduced compared to $w = 1$, demonstrating the algorithm's preference for less congested periods. Additionally, the algorithm shifts more requests to earlier (around minute 300) and later times (around minute 1400), avoiding congestion during the day and distributing them more to the morning or evening. At such low w values, the algorithm strongly favors uncongested periods, even at the cost of significant departure time deviations.

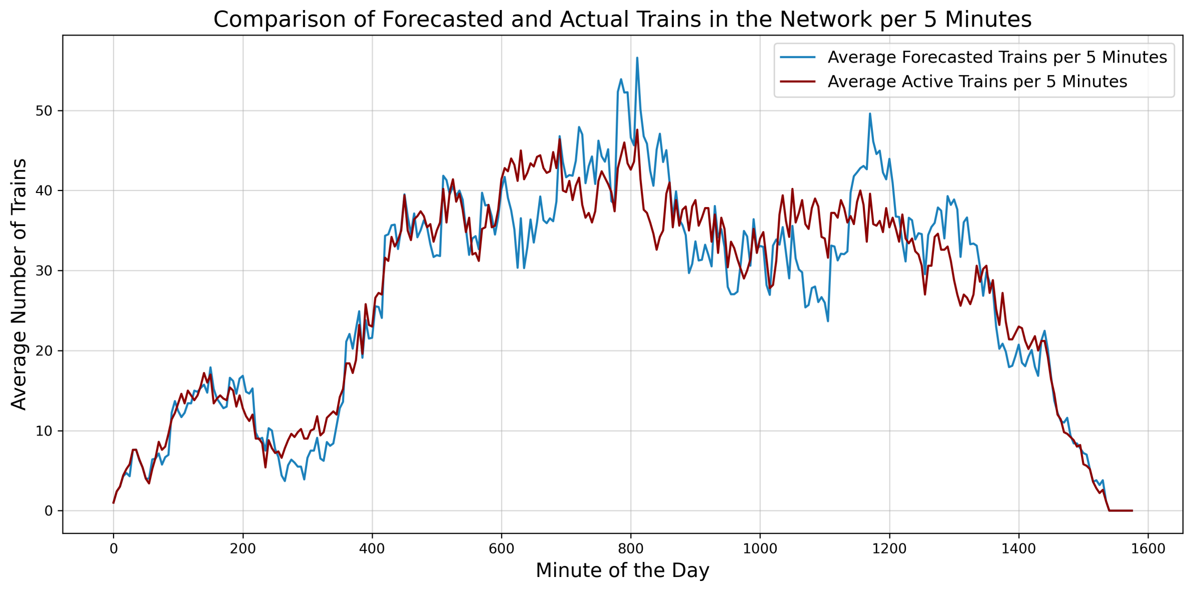


Figure 7.8: Average forecasted trains per 5 minutes for $w = 1$

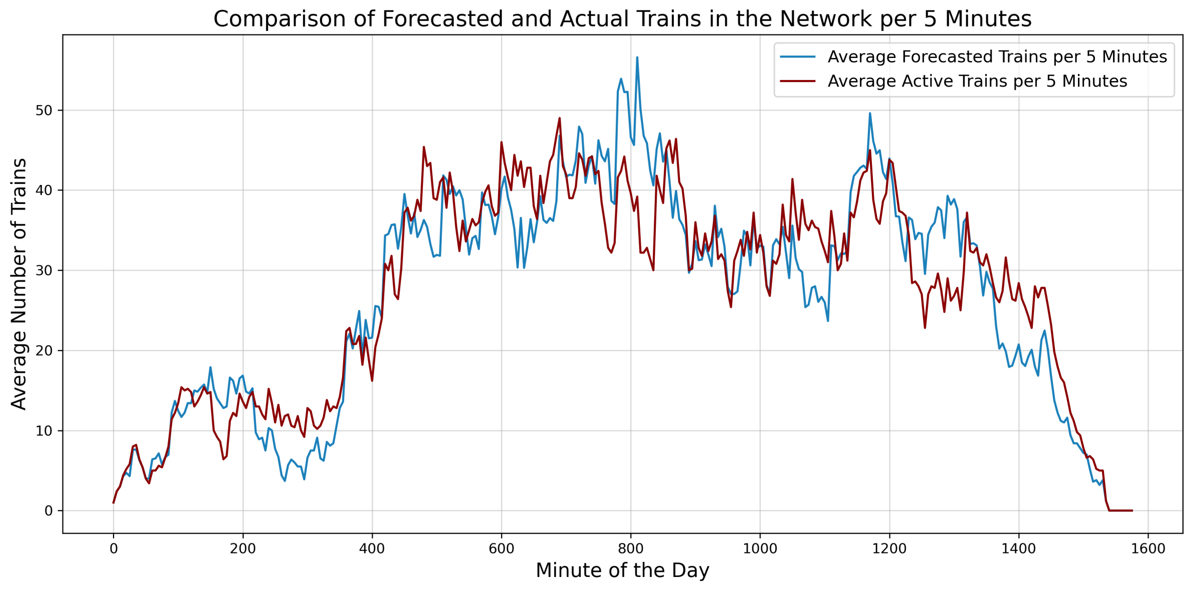


Figure 7.9: Average forecasted trains per 5 minutes for $w = 0.16$

Second Experiment

The second sensitivity analysis is conducted similarly to the one in Subsubsection 7.5.2.1 and evaluates the impact of reversing the order of requests on the scheduling model's performance. As in the first experiment, the capacity is set to 5 trains per hour, the congestion factor is squared, and it scales down per request. However, the request order is reversed to observe how this change affects the total (cumulative) time deviation and the squared time deviation for different values of w.

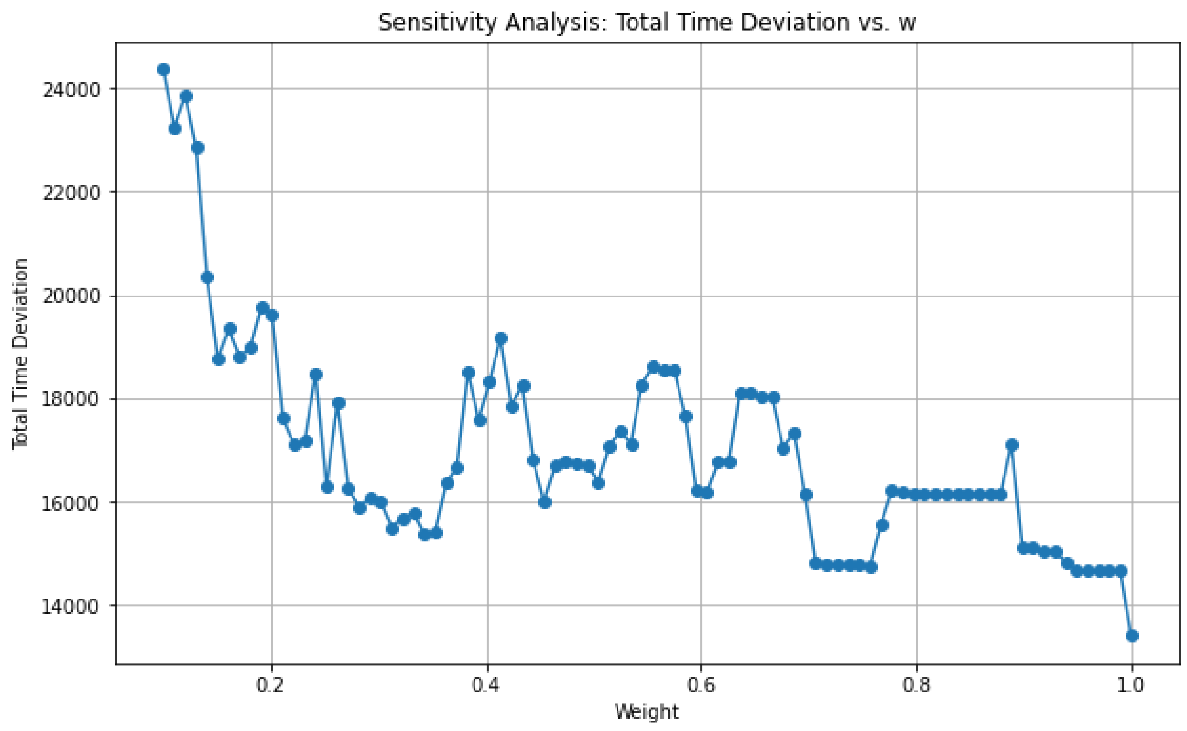


Figure 7.10: Total time deviation plotted per weight

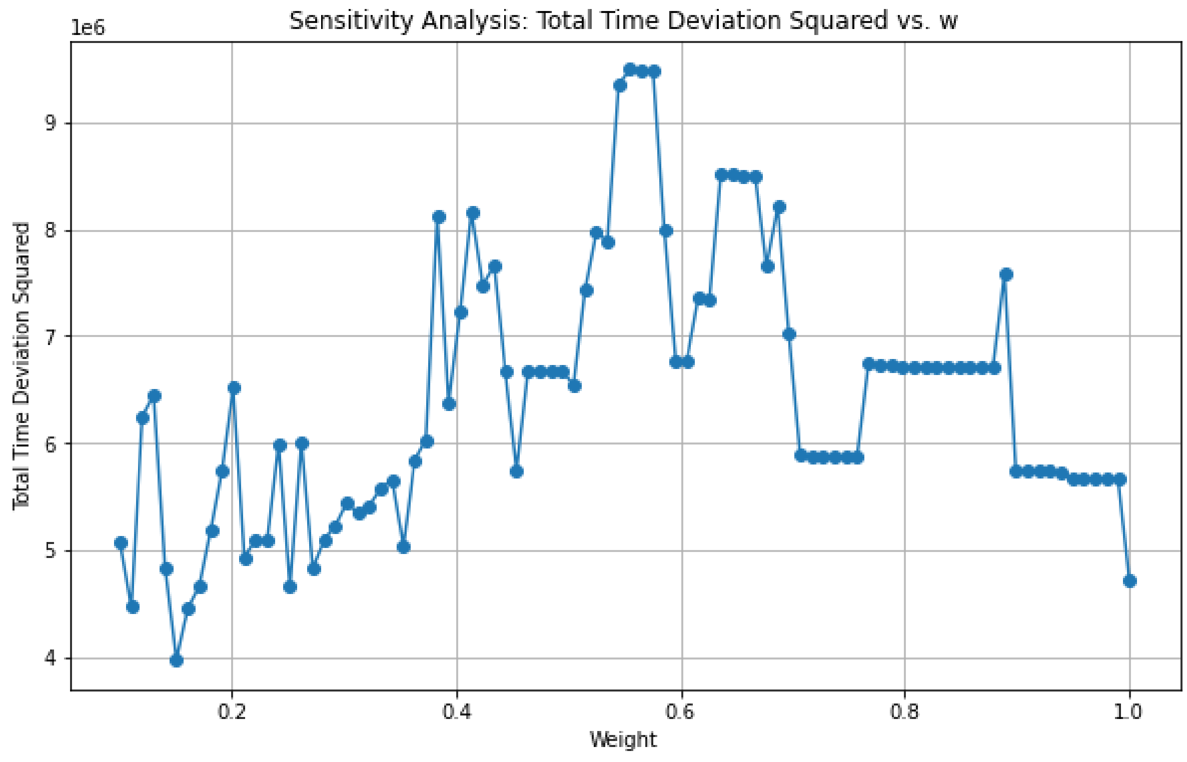


Figure 7.11: Total time deviation squared plotted per weight

Reversing the request order increases the total time deviation of the base case ($w = 1$) by approximately 1,500 minutes and the squared total time deviation by approximately 1.7 million. This highlights the significant influence of request order on scheduling outcomes. For smaller weight values ($w < 0.3$), minor improvements are observed in squared time deviations, but overall performance worsens compared to the base case. Two conclusions can be made: request order significantly influences scheduling outcomes for all w values, not just the base case, and in some scenarios, the algorithm may fail to achieve a performance improvement compared to the base case, for not a single value of w.

Table 7.8: Key Results for Second Experiment (Capacity = 5 Trains/Hour, Reversed Request Order)

Weight	Total Time Deviation (min)	Total Squared Deviation (min)	Number of Requests > 180 (min)	Observations
1.00	13,418	4,715,438	18	Base case
0.35	15,394	5,036,868	17	Balanced deviations
0.15	18,755	3,976,327	11	Best squared deviation

Table 7.9: Results for Reverse Request Order, Average and Maximum Time Deviation

Weight	Average Time Deviation (min)	Maximum Time Deviation (min)
1.00	17.43	896
0.35	19.99	897
0.15	24.50	851

The key results for $w = 1$, $w = 0.35$, and $w = 0.15$ are shown in Tables 7.8 and 7.9. From this data can be deducted that the algorithm reduces the number of extreme delays, with requests exceeding 180 minutes dropping from 18 in the base case ($w = 1$) to 11 at $w = 0.15$. Additionally, the maximum departure time deviation decreases slightly from 896 minutes at $w = 1$ to 851 minutes at $w = 0.15$.

However, the trade-offs are noticeable. As w decreases, the total time deviation increases significantly, from 13,418 minutes at $w = 1$ to 18,755 minutes at $w = 0.15$. Similarly, the average time deviation per request rises, reaching 24.50 minutes at $w = 0.15$, compared to 17.43 minutes in the base case. These results highlight the algorithm's prioritization of reducing extreme deviations, often at the expense of higher total and average deviations across all requests.

Figures 7.12, 7.13 and 7.14 visualize the scheduling changes for the selected weights in this reversed request order scenario.

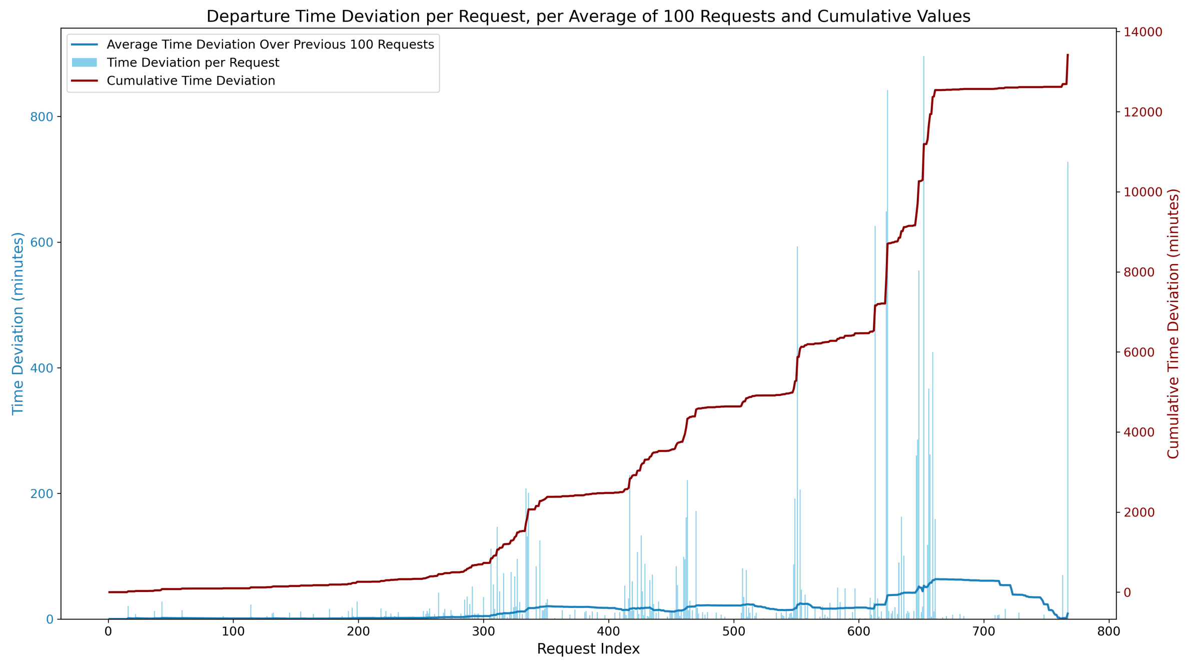


Figure 7.12: Time deviation per request, average time deviation per 100 requests and the cumulative time deviation for $w = 1$

The base case ($w = 1$) for reversed requests results in more extreme deviations, with 18 requests exceeding 180 minutes and higher total squared deviation compared to other weights.

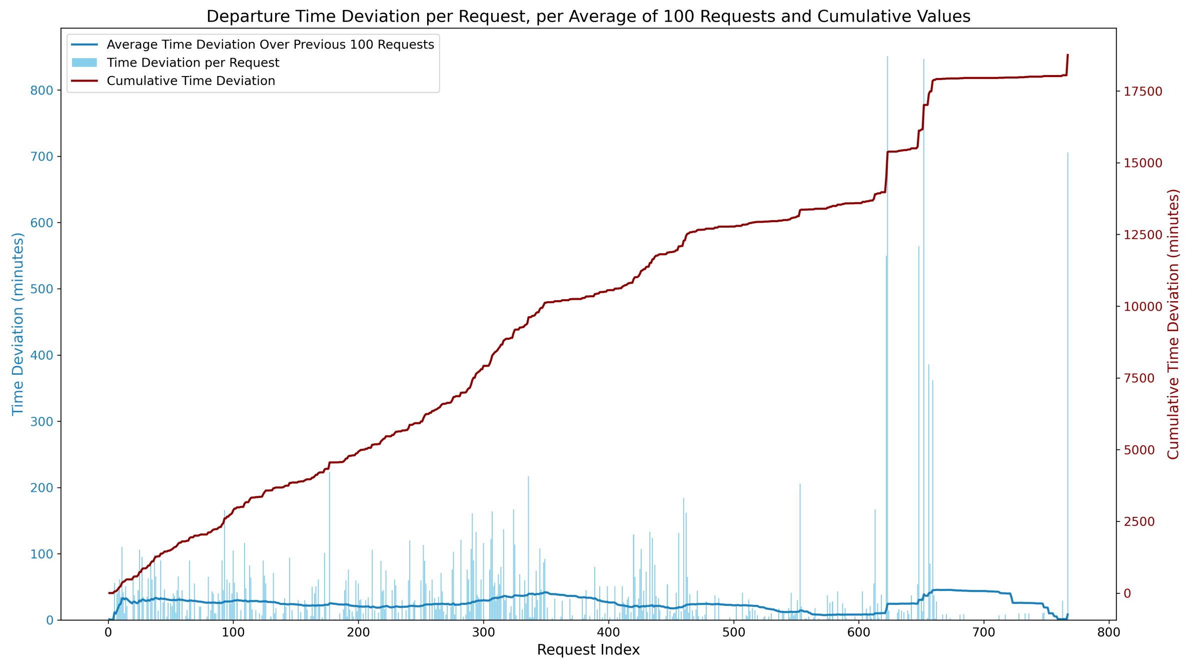


Figure 7.13: Time deviation per request, average time deviation per 100 requests and the cumulative time deviation for $w = 0.15$

For $w = 0.15$, the total time deviation increases significantly, but extreme deviations are reduced. Only 11 requests exceed 180 minutes, and most deviations are distributed across smaller delays.

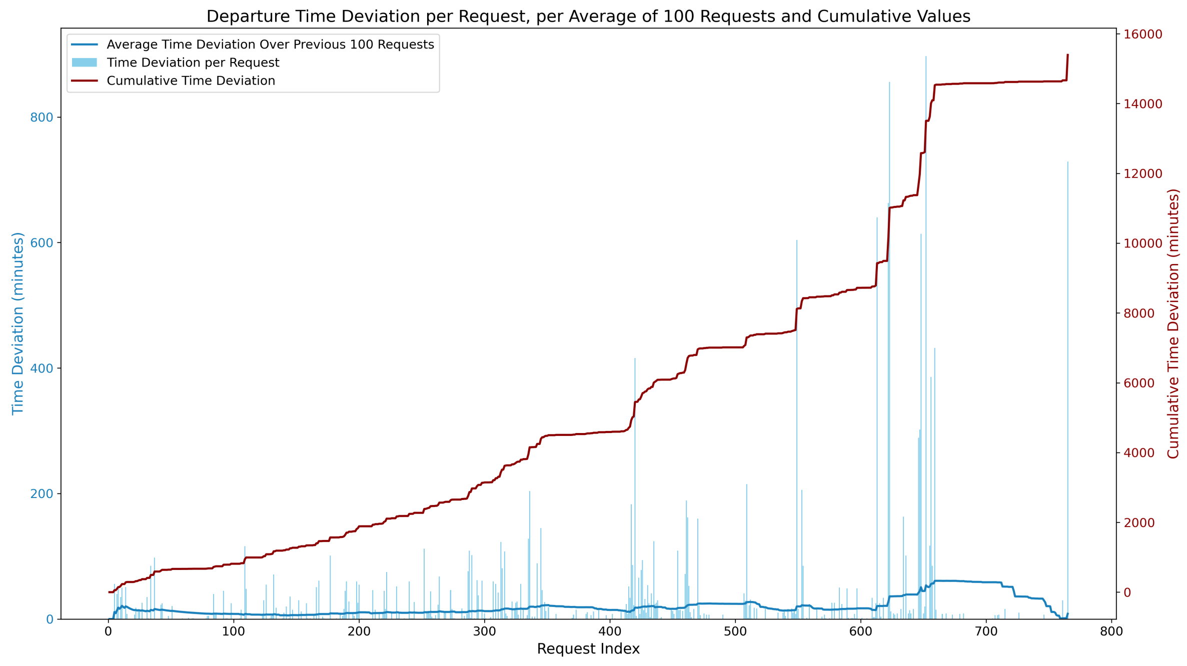


Figure 7.14: Time deviation per request, average time deviation per 100 requests and the cumulative time deviation for $w = 0.35$

For $w = 0.35$, the results balance smaller delays with fewer extreme deviations compared to the base case. This weight prioritizes a moderate trade-off between congestion and departure time deviations.

The load distributions for different w values are shown in Appendix E in Figures E.4 and E.5. While the distribution of requests differs slightly, the algorithm's behavior remains the same and no new conclusions can be drawn. When $w = 0.15$ the algorithm actively avoids forecasted congestion wherever possible, redistributing requests to less busy periods, often corresponding to the valleys in the graph.

Third experiment

The third sensitivity analysis evaluates the scheduling model in a high-capacity scenario. The network capacity is set to 10 trains per hour, with the congestion factor squared and scaling down over time. This experiment investigates how increased capacity affects the model's scheduling behavior, performance and consistency across different weights.

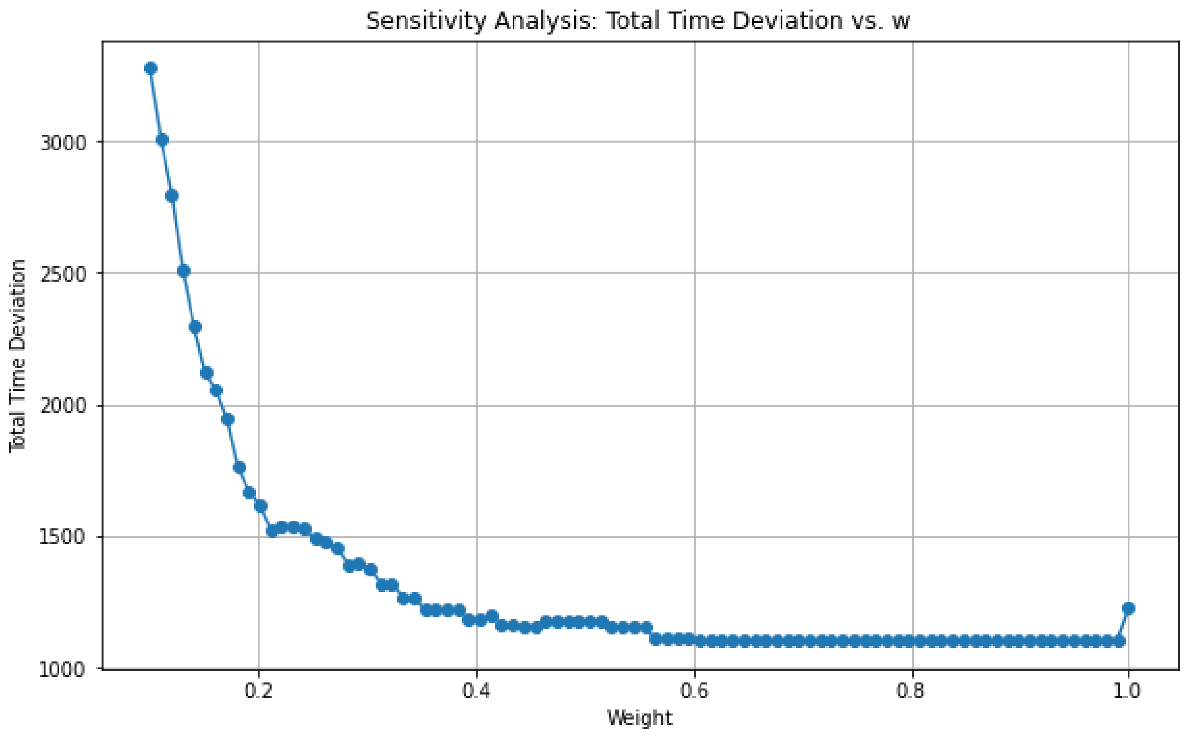


Figure 7.15: Total time deviation plotted per weight

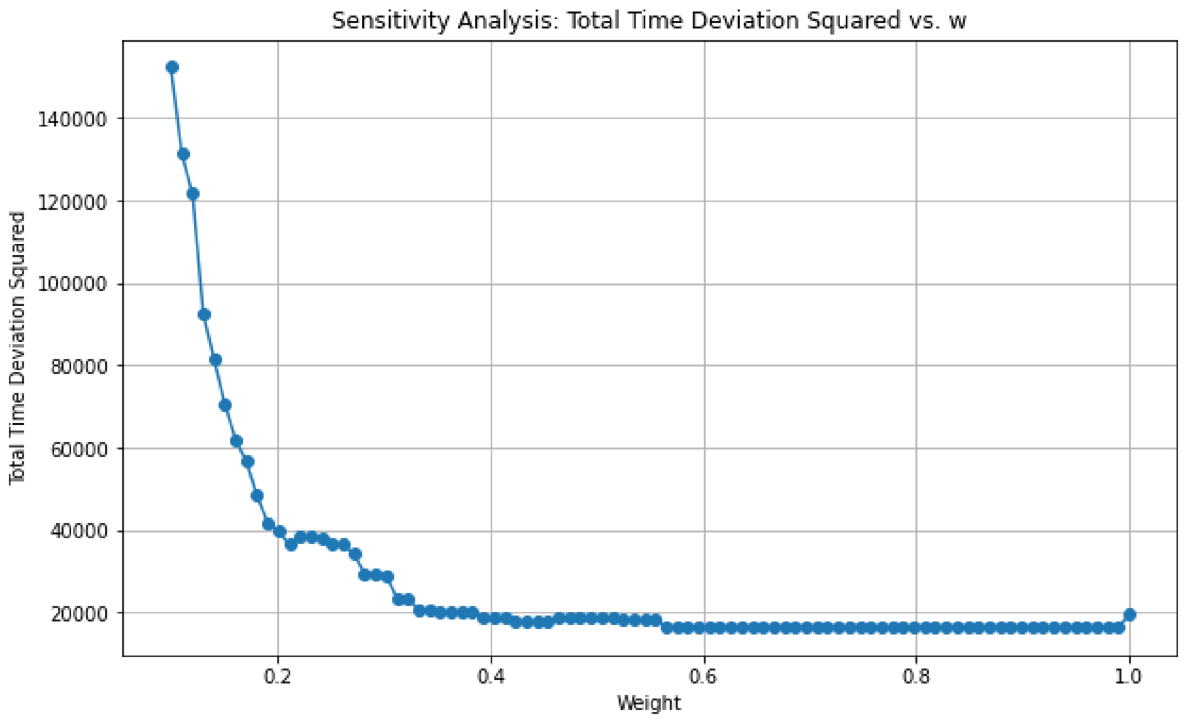


Figure 7.16: Total time deviation squared plotted per weight

The results in Figures 7.15 and 7.16 show a completely different behavior compared to the scenario with the lower capacity. Firstly, the total time deviation is reduced to 1,226 minutes for the base case scenario which is ten times smaller compared to the base case scenario where the capacity was 5 trains per hour. With more available capacity in the network, the scheduling model demonstrates a smoother trend across the weight range. Variations between weight values are less pronounced, as congestion penalties remain consistently low. The highest congestion factor across all itineraries is 0.718, compared to 2.874 in the low-capacity scenario. This means that congestion penalties do not play a big role in the scheduling decisions, as their impact is outweighed by time deviation penalties. The key results for the value w are showcased in Table 7.10.

Table 7.10: Results for Third Experiment (Capacity = 10 Trains per Hour)

Weight	Total Time Deviation (min)	Total Squared Deviation (min)	Number of Requests > 180 (min)	Observations
1.00	1,226	19,422	0	Base case
0.6-0.99	1,103	16,465	0	Minimal deviations

Table 7.11: Results for Normal Request Order (Capacity is 10 trains per hour)

Weight	Average Time Deviation (min)	Maximum Time Deviation (min)
1.00	1.65	60
0.6-0.99	1.44	53

The key results for $w = 1$ and $w = 0.6\text{-}0.99$ are shown in Tables 7.10 and 7.11. These results demonstrate the reduced influence of the algorithm in high-capacity scenarios where the network can accommodate most requests with minimal deviation. For both weights, no requests exceed 180 minutes, highlighting the absence of extreme delays.

At $w = 0.6\text{-}0.99$, the total time deviation decreases slightly to 1,103 minutes compared to 1,226 minutes in the base case ($w = 1$). The squared total time deviation also shows a notable improvement, reducing from 19,422 to 16,465 minutes. Additionally, the average and maximum time deviations are marginally lower, with the average deviation dropping from 1.65 minutes to 1.44 minutes and the maximum departure time deviation from 60 minutes to 53 minutes.

These results indicate that, in high-capacity networks, the algorithm performs consistently well regardless of weight, with minimal differences in outcomes. The high capacity ensures that most preferred departure times can be accommodated without significant adjustments, leading to stable and predictable scheduling results.

The scheduling behavior for the selected weights in Table 7.10 are visualized below to demonstrate the scheduling decisions being made in the model under high-capacity conditions.

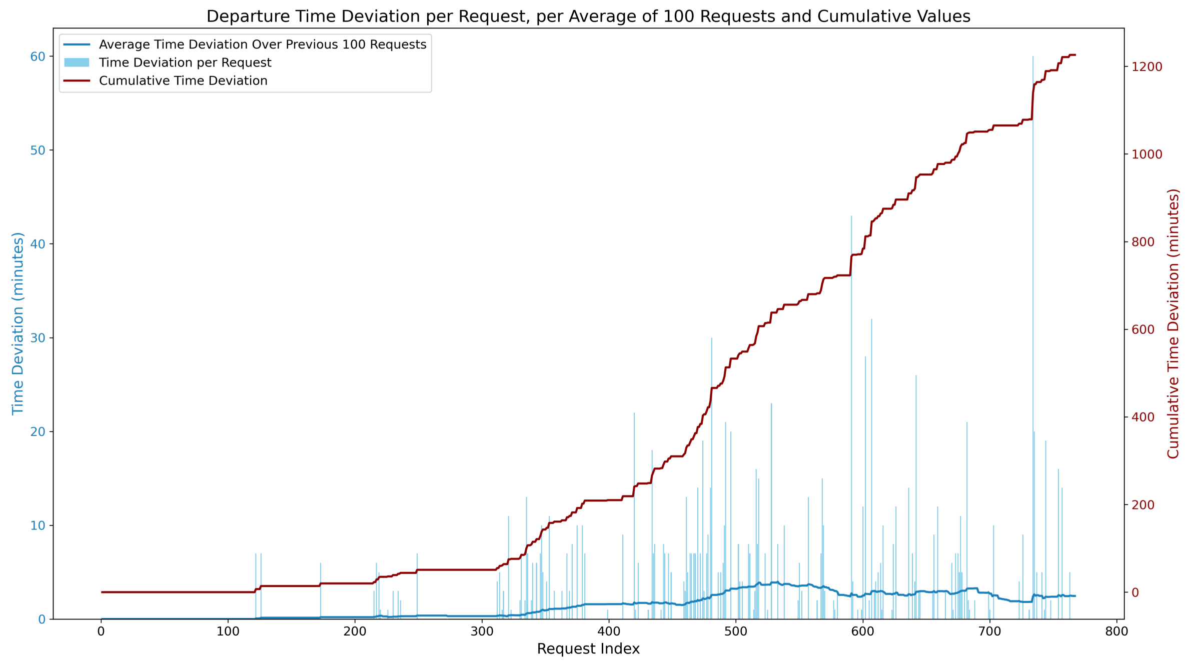


Figure 7.17: Time deviation per request, average time deviation per 100 requests and the cumulative time deviation for $w = 1$

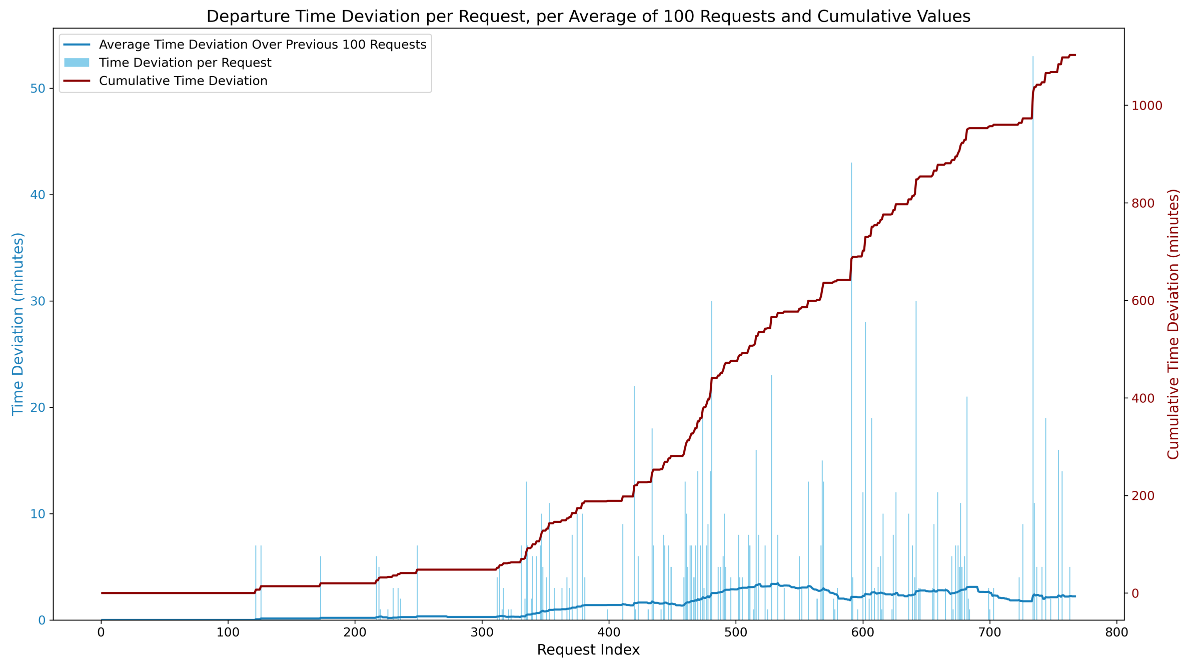


Figure 7.18: Time deviation per request, average time deviation per 100 requests and the cumulative time deviation for $w = 0.9$

No requests experienced a time deviation exceeding 180 minutes, with the maximum time deviation being only 43 minutes. This contrasts the high-capacity scenario, where extreme deviations (of multiple hours) frequently occurred.

When looking closely at the scheduling process, the results are slightly better when a value for w is chosen between 0.60 and 0.99, resulting in a decrease of 123 minutes. The difference between Figure 7.17 and 7.18 is not as noticeable compared to the low capacity scenario in experiment one 7.5.2.1

and two 7.5.2.2. To put things in perspective, the average time deviation per request is approximately equal to one and a half minute. The biggest time deviation of 43 minutes can also be considered as acceptable when looking at the scheduling process. One could argue that in this scenario it is not really needed to start actively rescheduling requests. Everything fits easily in the network.

No requests experienced a time deviation exceeding 180 minutes, with the maximum deviation being only 43 minutes. This marks a stark contrast to the low-capacity scenario, where extreme deviations of several hours frequently occurred. A closer analysis of the scheduling process reveals slightly better results when w is chosen between 0.60 and 0.99, reducing the total time deviation by 123 minutes. However, the difference between Figures 7.17 and 7.18 is minimal compared to the more pronounced variations observed in the low-capacity scenarios of experiments one 7.5.2.1 and two 7.5.2.2. On average, the time deviation per request is approximately one and a half minutes, with the largest deviation of 43 minutes being considered acceptable. Given the high capacity of the network, active rescheduling or offering alternative itineraries seems unnecessary, as requests are easily accommodated without significant delays. The weight parameter (w) does not have a big positive impact on the scheduling model and when it is lowered too much, the outcomes become worse compared to the base case.

The load distributions for a network capacity of 10 trains per hour show that forecasted demand and the actual allocated requests are nearly identical, as can be seen in Appendix E in Figures E.3 and E.2. With increased capacity, most preferred departure times are met, resulting in minimal differences between the forecasted and actual graphs. No significant variations are observed across different w values, except when w is reduced to extremely low levels.

Conclusions Sensitivity Analysis
Using the KPI's mentioned in Table 1.2, the results of the sensitivity analysis can be quantified and therefore evaluate the performance of the scheduling model by comparing the results to the base case.

The cumulative departure time deviation for the high capacity scenario remains relatively stable across most values of w, with minor improvements or degradations compared to the base case. However, when w falls below a certain threshold, the cumulative departure time deviation increases rapidly, but consistently. For low-capacity scenarios, the effect of w is much more pronounced and chaotic, with no clear pattern of improvement or degradation. Performance fluctuates significantly for each value of w, and results are also highly dependent on the request order (for the base case and for values of w). This indicates a greater sensitivity of the model to the tuning of the weight parameter under constrained network conditions.

The cumulative departure time deviation squared shows slightly different behavior. In low-capacity scenarios, lower w values tend to reduce extreme deviations better, but sometimes at the expense of increasing total and average deviations. Also, the behavior is still chaotic and not consistent for different values of w. For high-capacity networks, the improvements are marginal, with reductions in extreme deviations being negligible due to the relatively small maximum departure time deviations already present (e.g., a reduction from 60 to 53 minutes).

The load distribution shows how the algorithm reallocates requests to avoid busy periods, shifting them to forecasted less busy times. In the base case, requests are scheduled without avoiding congestion, resulting in the actual load and forecasted load are kept similar if possible unless no capacity is left. While for lower w values, the algorithm actively steers requests away from busy periods, prioritizing less congested times.

Another key aspect is the distribution of offering departure time deviations. For lower w values, alternatives are offered earlier in the time horizon, evenly spreading departure time deviations throughout. In contrast, the base case first schedules requests until capacity is reached, with alternatives primarily offered later in the horizon, concentrating deviations near the end.

Analyzing average and maximum departure time deviations provides additional context. In high-capacity scenarios, improvements in average deviations are minimal, such as an average reduction of 13 seconds per request from 1.65 to 1.44 minutes. In low-capacity scenarios, the differences are more pronounced, with variations of several minutes on averages ranging from 15 to 18 minutes per request. Maximum deviations, however, show significant reductions for low-capacity scenarios, with decreases of up to 719 minutes (from 1003 to 284 minutes). However, for the second experiment with the reversed request order the reduction is at most 45 minutes compared to the base case showing that sometimes it is difficult to get rid of extreme outliers in departure time deviation. In contrast, improvements in maximum deviations are modest for high-capacity scenarios, such as a reduction of just 7 minutes (from 60 to 53 minutes).

The weight parameter w fundamentally drives a trade-off between minimizing future congestion and minimizing the departure time deviation for the request currently being handled. This trade-off is more challenging to manage in low-capacity scenarios, where small changes in w produce significant differences in outcomes. High-capacity networks, on the other hand, exhibit greater stability, as the surplus capacity reduces the sensitivity of the results to w. The request order also plays a critical role, especially in low-capacity scenarios, influencing the model's ability to improve performance relative to the base case. The base case ($w = 1$) acts as a greedy algorithm that performs inconsistently, sometimes yielding better results and other times being outperformed by smaller values of w.

It is important to clarify that the results are driven not by capacity alone, but by forecasted congestion, which depends on both capacity and forecasted demand. In the experiments conducted the amount of requests coming in stays the same and therefore it seems like capacity is influencing the outcomes, but it is the combination of capacity and demand.

These results reveal that the optimal value of w is highly scenario-dependent, making it difficult to generalize a single best value for all conditions. This raises an important question for further investigation: Can a repeated sensitivity analysis across multiple request orders identify an average value of w that consistently improves scheduling outcomes compared to the base case? This question motivates the next section, which explores the consistency and reliability of values for w through repeated testing.

7.5.3. Repeated Sensitivity Analysis

This section evaluates the impact of repeated sensitivity analysis to identify a consistent and optimal value of the weight parameter (w) for the scheduling model. The experiments aim to address the variability observed in previous sensitivity analyses by testing the model across multiple randomized request orders. The objective is to determine whether a particular value of w consistently outperforms the base case scenario ($w = 1$) across various network configurations and parameter settings.

Experimental Setup

The repeated sensitivity analysis explores how different values of w influence the scheduling model's performance across 12 distinct experiment configurations. Each experiment combines variations in four key parameters:

1. **Objective function:** Static congestion factor or CF decreasing per request.
2. **Congestion factor per section:** Linear or squared congestion factors for each section.
3. **Request order:** Randomized over 50 iterations for each configuration.
4. **Capacity:** 5, 7.5, 10 trains per hour.
5. **Weight:** $w = 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 0.99, 1$

Each experiment evaluates the value of w against the base case ($w = 1$), which minimizes time deviation without considering the congestion factor.

Objective Function: The congestion factor is either static or decreases linearly as requests are processed. This approach prioritizes offering alternatives earlier in the planning horizon, reserving capacity for later requests. Testing these two scenarios assesses whether request-dependent congestion factors improve scheduling outcomes. This is further discussed in Subsection 7.4.3.

Congestion factor per Section: As described in Subsection 6.1.2, the congestion factor per section is modeled as either linear or squared. Squared factors penalize heavily congested sections more, potentially improving outcomes in high-demand scenarios. The experiments compare these configurations to determine which approach produces better results.

Request Order: From the experiments conducted in Subsection 7.5.2 the importance of the order of the requests is highlighted. This is why the order in which requests are processed is randomized over 50 iterations to evaluate its impact on scheduling outcomes. Since allocation decisions significantly depend on request order, this test measures the variability in results and how w performs under different sequences.

Capacity: Experiments test network capacities of 5, 7.5, and 10 trains per hour to analyze the model's adaptability under different levels of congestion. Higher capacities are expected to reduce sensitivity to w, while lower capacities may highlight the weight's importance in managing congestion.

Weights: To manage computational demands (approximately 3 hours per experiment), the weights tested are limited to $w \in 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 0.99$. Weights below 0.2 are excluded due to their disproportionate impact on deviations, which distorts result interpretation. The inclusion of $w = 0.99$ creates the scenario where the model chooses between itineraries with the same departure time, but the congestion factor differs.

Performance Metrics
The following metrics are used to evaluate the results:

- Average Improvement in Total Time Deviation: Difference in total time deviation compared to the base case, averaged over 50 iterations.
- Average Improvement in Squared Time Deviation: Similar to the above, but penalizing larger deviations more heavily by squaring them.
- Percentage of Better Results (Total Deviation): Proportion of experiments where a weight resulted in a lower total deviation than the base case.
- Percentage of Better Results (Squared Deviation): Similar to the above but applied to squared deviations.

Baseline Scenario: The baseline scenario corresponds to $w = 1$, where requests are scheduled greedily without prioritizing future flexibility. A weight is considered effective if it consistently outperforms the base case in terms of keeping the average time deviation close to the same value while having a smaller average time deviation squared.

7.5.4. Results and Discussion Repeated Sensitivity Analysis

The repeated sensitivity analysis provides insights into the behavior of the scheduling model under varying conditions. Adjusting the weight (w) and other parameters followed by running the experiment 50 times, reveals how the model performs on average for a specific scenario. By conducting these experiments the question will be answered on whether there is a scenario of a combination of parameters chosen, where the model will consistently perform better compared to the base case scenario.

In this Subsection the key takeaways from the results are discussed, based on all the experiments conducted for the repeated sensitivity analysis. The complete results can be found, visualized by the use of box plots, in Appendix A and all the results from showing the improvement per scenario in numbers are gathered and organized in Tables shown in Appendix B.

Behavior of Changing the Weight
The results, visualized in Appendix A and B, show that reducing w from 1 to 0.99 produces minimal changes compared to the baseline. When $w = 0.99$, the model prioritizes choosing the lowest congestion factor for itineraries with identical departure time deviations. While this is expected to improve departure time deviations, the results show no consistent significant improvement. This could be due to the limited availability of alternative paths between origin-destination pairs, reducing the impact of congestion factor prioritization. In most scenarios, the results for $w = 0.99$ are similar to those for $w = 1$, with no noteworthy differences. However, some specific scenarios, such as B.1, show a slight improvement in average departure time deviations, but still a decrease in the time deviation squared. These somewhat small improvements can be marked as negligible due to their small size and lack of consistency.

A more noticeable trend occurs as w decreases further. The performance of the model stays comparable, with optimal results sometimes being observed around $w = 0.9$. Decreasing w further, the results show a slight increase in total time deviations before improving again for weights in the range of $w = 0.4$ to $w = 0.6$. This behavior can be explained by the penalty trade-off between congestion factor and time deviation. For w values near 0.4-0.6, the penalty for a congestion factor change becomes favorable when offering a 60-minute departure time deviation. Table 7.12 illustrates this trade-off, where a lower weight makes the second path option with the time deviation of 60 minutes better due to its lower congestion factor.

Table 7.12: Example of departure time deviation and congestion factor for itineraries of two paths

Departure Time Deviation (minutes)	Congestion Factor (Path 1)	Congestion Factor (Path 2)
0	1.8	1.3
60	1.6	0.7

As w decreases below 0.4, the total (cumulative) time deviation increases sharply. At this point, the model excessively prioritizes minimizing the congestion factor, often neglecting the importance of minimizing time deviations. This behavior leads to the frequent offering of alternative itineraries with larger time deviations, undermining overall scheduling performance.

The model behaves mostly as expected, but the negligible impact of reducing w slightly (e.g., $w = 1$ to $w = 0.99$) is unexpected. Potential reasons could be due to inaccuracies in demand forecasting (Subsection 5.1.1) which is not done perfectly and could be elaborated on by extending the calculations for the utility per path.

Or the negligible impact could be due to the averaging of congestion factors across sections in an itinerary. Dividing the congestion factor by the number of sections in an itinerary is intended to prioritize routes with lower average congestion. This approach generally leads to better scheduling decisions, as it encourages the selection of itineraries that impose less strain on the network. However, this method can also inadvertently lead to suboptimal outcomes.

For instance, by focusing on minimizing the average congestion factor, the model might prioritize itineraries with more sections, each having a low congestion factor, over shorter routes with slightly higher congestion factors. While this reduces the average congestion, it results in the unnecessary use of additional sections. Consequently, this behavior can increase the total number of sections occupied in the network, consuming more capacity than needed.

As the scheduling horizon progresses, this inefficient allocation of capacity can lead to an overcrowded network. The reduced availability of free sections for later requests results in higher time deviations or even the inability to schedule requests optimally. Thus, while dividing the congestion factor by the number of sections generally aligns with the goal of reducing congestion, it can unintentionally strain network congestion and compromise scheduling efficiency in cases where longer routes are selected unnecessarily.

Squaring the Congestion Factor or not

The results demonstrate that squaring the congestion factor generally enhances performance, particularly in low-capacity networks (5 trains per hour). Squaring amplifies the penalty for heavily congested sections, encouraging the model to allocate requests more efficiently. This trend is consistent for both request-dependent and request-independent scenarios.

In moderate-capacity networks (7.5 trains per hour), squaring the congestion factor still improves results in most cases but exhibits less consistent advantages. The average congestion factor in this scenario is 0.230, and the maximum congestion factor among all itineraries is 1.466. Compared to low-capacity networks, where the average congestion factor is 0.323 and the maximum is 2.874, the reduced variability in congestion lessens the benefits of squaring. In some cases, linear congestion factors perform slightly better due to the more balanced congestion levels.

In high-capacity networks (10 trains per hour), the impact of squaring the congestion factor diminishes significantly. The average congestion factor is only 0.161, with a maximum of 0.718, making the penalty differences between linear and squared congestion factors negligible. Squaring remains effective when congestion variability is high, as seen in low-capacity scenarios, but its influence diminishes as capacity increases and network congestion becomes less of a problem.

Congestion Factor Request Dependent or Not
The results show minimal differences between request-dependent and request-independent congestion factor scenarios. In request-independent scenario's, the congestion factor penalty remains constant throughout the scheduling horizon, leading to penalties coming off stronger compared to the request dependent penalty. This is seen clearly when comparing the time deviation results of the lower weights (e.g., $w = 0.2$).

While the weights do not behave identically across the two approaches, neither method consistently outperforms the other. Scaling the congestion factor down per request, slightly outperforms request-independent scenario, but not convincingly. This indicates that both could be viable strategies, with their effectiveness depending more on the specific network conditions and weights than on the scaling method itself.

Performance Under Different Capacities
Changes in capacity significantly affects the model's performance, primarily through its impact on scheduling flexibility and congestion factor values.

From the Subsection 7.5.1, the results show that for low-capacity networks (5 trains per hour), scheduling of the requests becomes more chaotic and unpredictable. In such cases, the range of w values where the total time deviation improves compared to the baseline becomes smaller and less consistent. Small changes in w heavily influence total time deviations, with optimal results for weights ranging between $w = 0.8$ and $w = 0.99$. However, these improvements are marginal, averaging around 100 minutes out of a total deviation of 13,000 minutes, and squared deviations always perform worse. Surprisingly, the algorithm's ability to tactically reschedule requests does not yield significant gains in these scenarios. It would be expected that in the cases of limited capacity, the model could provide a better outcome when applying the model. Additionally, time deviations vary widely when conducting the experiment 50 times, ranging from 11,000 to 19,000 minutes across iterations for $w = 1$, showing the impact on the scheduling outcomes from the order in which requests come in.

In moderate-capacity networks (7.5 trains per hour), results fluctuate less compared to the low-capacity network, with deviations ranging between 2,250 and 3,100 minutes for $w = 1$ after 50 experiments conducted. While weights between $w = 0.8$ to $w = 0.99$ still perform optimally, they generally offer no significant advantage over $w = 1$.

In high-capacity networks (10 trains per hour), the model becomes less sensitive to w, as most requests are easily accommodated. Weights between $w = 0.7$ and $w = 0.99$ show slight advantages, but the improvements are negligible, averaging around an improvement of one second per request. This highlights the reduced importance of congestion minimization in networks with a high capacity. The range of time difference for 50 iterations is between 1,100 and 1,400 minutes for $w = 1$.

Conclusion Repeated Sensitivity Analysis
Based on the repeated sensitivity analysis the results can be presented based on the last two KPI's presented in Table 1.2 referring to the average cumulative departure time deviation and the average cumulative departure time deviation squared.

The repeated sensitivity analysis reveals that the optimal weight w for the scheduling model depends significantly on the amount of requests, network capacity, request order, and parameter configurations. In low-capacity networks, scheduling outcomes are highly sensitive to w, with slight improvements observed in average cumulative departure time deviation and the average cumulative departure time deviation squared for weights around $w = 0.8$ and $w = 0.99$. However, these improvements are marginal and inconsistent, highlighting the unpredictable nature of scheduling under constrained capacity.

As capacity increases (while the amount of requests stays the same), the model's sensitivity to w decreases. In moderate-capacity network there is no significant advantage over the base case scenario regarding the average deviation and average deviation squared. In high-capacity networks, the model becomes largely indifferent to w, as most requests are easily accommodated, and congestion minimization has minimal influence on results.

The analysis also shows that squaring the congestion factor generally improves performance in low-capacity networks by penalizing heavily congested sections more effectively. However, this benefit decreases as capacity increases, with negligible differences observed in high-capacity scenarios. The decision to scale congestion factors per request offers slight advantages in some cases but does not consistently outperform static penalties.

As already concluded in the conclusion from Section 5.4.5, the order of the requests has a significant impact on scheduling outcomes, particularly in low-capacity networks. Randomizing request orders introduces such a level of variability that, even though the requests are the same, the order completely changes the scheduling and thus the results on total time deviations. This difference is highlighted by the top and bottom of the box plot indicating the maximum and minimum cumulative deviations.

In summary, no single value of w consistently improves results across all scenarios. While weights in the range of $w = 0.7$ to $w = 0.99$ provide better results in some cases, their effectiveness depends on the network's capacity and conditions. Further exploration is needed to determine if an average optimal weight can be identified for consistent performance under varying scenarios.

7.6. Conclusion Case Study

By using the KPI's presented in Table 1.2, the performance of the algorithm can be measured and evaluated by comparing the results to the results of the base case (which simulates the first come first served approach currently used by ProRail). The comparative analysis demonstrates that the algorithm's performance relative to the base case is highly dependent on the demand, network capacity, request order, and the weight parameter w.

In high-capacity networks, the algorithm offers minimal improvements over the base case, as excess capacity (with similar demand inputs) reduces the influence of both congestion and weight adjustments. Cumulative departure time deviations and their squared values remain stable (unless significantly lowering w), and improvements in maximum deviations are modest. Conversely, in low-capacity scenarios, the algorithm shows greater potential, reducing maximum departure time deviations by up to 719 minutes in some cases and showing significant improvements in the cumulative time deviation squared indicating that extreme time deviations are less frequent. However, by conducting the repeated sensitivity analysis results of improvement remain inconsistent, with performance of the algorithm being heavily influenced by request order and weight tuning.

The algorithm's ability to actively redistribute requests away from congested periods early on in the time horizon, demonstrates an improvement in offering alternatives more evenly throughout the time horizon as the base case mostly offers alternatives near the end of the horizon.

Ultimately, while the algorithm outperforms the base case in certain scenarios, particularly in managing extreme deviations in low-capacity networks, its effectiveness varies significantly and inconsistently. The base case ($w = 1$) proves surprisingly effective in many scenarios, behaving as a robust greedy algorithm. No single configuration consistently delivers better results, highlighting the need for further investigation to identify robust, scenario-independent parameter settings.

8 Conclusion, discussion and recommendation

This chapter presents the conclusion, discussion, and recommendations based on the findings of this thesis. The conclusion answers the main research question and its subquestions. The discussion highlights potential limitations and situates the findings within existing literature. Lastly, recommendations are made for improving ProRail's scheduling approach and for future research directions.

8.1. Conclusion

The objective of this research was to develop and evaluate an alternative scheduling algorithm which tries to minimize forecasted congestion while also minimizing departure time deviations, with the goal of reducing the number of extreme time deviations while keeping the overall departure time deviation similar. The objective comes from the main research question: *"How can a dynamic scheduling model that balances minimizing forecasted congestion and departure time deviations per customer request, improve rail freight scheduling in the Netherlands?"* which is answered through the use of multiple subquestions. The first subquestion being:

What are the limitations of the current scheduling approach used by ProRail, and how do they impact customer satisfaction?

ProRail's first-come, first-served approach favors requests being handed in early, which sometimes lead to fewer options for later submissions. While ProRail almost always manages to plan requests into the network, the departure time deviations from customers' preferred schedules can vary significantly, sometimes by multiple hours or even days. This creates uncertainty for late-submitters, who are disproportionately impacted by these deviations.

To avoid such outcomes, customers now secure itineraries early "just in case," even without the certainty of using them. This behavior results in unnecessary cancellations or rescheduling of some requests (sometimes up to twenty times), adding inefficiencies to the planning process. Additionally, the reliance on a manual, trial-and-error approach for allocation limits ProRail's ability to explore optimal solutions, further underutilizing the network's capacity. This manual scheduling process is also unnecessary, as automation could significantly reduce the amount of labor required while improving efficiency and consistency in planning. These issues ultimately result in lower customer satisfaction, as significant deviations from requested schedules reduce trust in the scheduling process.

How can a deterministic demand forecasting model be created using request data, and how can this information be integrated into the scheduling model?

A deterministic demand forecasting model was constructed using request data on origins, destinations, and preferred departure times. However, the challenge lies in the fact that from this request data alone, it is not possible to know how busy specific paths or sections of the network will become. This is because path choices for individual requests have not yet been made, and overlapping paths or shared sections between different origin-destination pairs further complicate the forecasting process.

To solve this problem, the C-Logit model was employed to distribute demand across feasible paths between origin-destination pairs. This model accounts for overlapping paths by assigning probabilities based on path attributes, in this case total overlap distance. Once demand is divided among paths, travel times are used to allocate the demand over time to the corresponding sections.

By examining individual network sections, the demand for each section is calculated by summing the demand from all paths that traverse it. Using this section-specific demand, along with the available and total capacity, a congestion factor is calculated for each section. This factor provides an approximation of how busy the section is expected to become, offering a predictive measure for potential bottlenecks.

Once congestion factors for individual sections are determined, they are used to calculate the total congestion factor for an itinerary, which consists of multiple sections. For each itinerary, the congestion factors of all sections are summed and then divided by the number of sections to compute an average congestion factor. Two models were tested: one using the normal congestion factors and another squaring the congestion factors for each section before summing. The results from the case study indicate that squaring the congestion factor provides better performance by penalizing heavily congested sections more.

In summary, the deterministic demand forecasting model uses the C-Logit framework to estimate demand per path and from this compute section-level congestion factors. These values are then integrated into the scheduling model, enabling it to prioritize less congested paths.

How can a mathematical model be developed to dynamically schedule freight train requests while balancing the two objectives on forecasted congestion and departure time deviation?

A mathematical model is constructed to replicate ProRail's current scheduling approach by minimizing departure time deviations per request. A second objective, minimizing congestion factors per request, was added to the objective, allowing the focus to shift between the two objectives using a weight parameter (w).

The mathematical model, including its objective function and corresponding constraints, was developed by thoroughly analyzing ProRail's current scheduling system, defining sets, parameters, and variables, and making certain justified assumption for simplification. The disparity in behavior between time deviations and congestion penalties makes the model's outcomes inconsistent and hard to predict. From the mathematical model a sensitivity analysis was conducted to test various weights as inputs and identify the optimal trade-off from these results. While the model captures the scheduling dynamics of the system effectively, it struggles to find a balance between the two objectives, not knowing which should be prioritized more and by how much.

How does the proposed dynamic scheduling model perform compared to the current approach?

The primary objective of this research was to minimize total time deviation squared while keeping total time deviation similar or smaller. By comparing the outcomes of these measurements, the aim is to reduce extreme departure time deviations, which are generally less acceptable to customers. Other factors influencing customer satisfaction such as costs, travel time, or distance were excluded, focusing solely on time deviation.

Experiments show that, when all requests and its order are known, an optimal weight (w) can outperform the current method by reducing extreme time deviations in size and quantity. However, the best value for w is inconsistent across different scenario's. This inconsistency shows the model's unreliability and whether a single w can consistently improve outcomes.

Testing ten different values for w, across fifty different request orders, under various parameters (e.g., squaring congestion factors, capacity variations,) revealed that no value for w consistently produced better results referring regarding the KPI's presented in the introduction. Especially when also taking into account that the deterministic forecasting model used in this research lacks the uncertainty present in real-world demand forecasting. Including such variability would likely increase fluctuations even more, making the model's performance even less consistent.

One of the most notable improvements achieved by the algorithm is in the distribution of alternative offers. In the base case scenario, alternatives are primarily offered near the end of the scheduling horizon, which disadvantages customers submitting requests later. By reducing the value of w, the algorithm offers alternatives more evenly throughout the entire scheduling period. This improvement could increase customer trust in the system, as customers are less likely to perceive the scheduling process as biased against late requests. A more balanced distribution of alternatives might discourage customers from submitting requests prematurely "just in case," potentially leading to fewer cancellations or last-minute changes. Such improvements could result in reduced administrative workload for ProRail and enhance overall scheduling efficiency.

"How can a dynamic scheduling model that balances minimizing forecasted congestion and departure time deviations per customer request, improve rail freight scheduling in the Netherlands?"

In conclusion, while the proposed algorithm reduces extreme departure time deviations in some scenarios, it does not reliably improve overall performance compared to the current scheduling approach. The results demonstrate that balancing forecasted congestion with time deviation is theoretically achievable, but highly sensitive to network conditions, request order, and weight selection. The lack of consistency in results and the dependence on deterministic forecasting limit the algorithm's practical value in its current form.

Nevertheless, the research provides valuable insights into objective balancing and dynamic scheduling methodologies. It highlights the challenges of combining minimizing departure time deviation and forecasted congestion, revealing the sensitivity of outcomes to parameter adjustments and request order. Furthermore, the analysis of congestion factor methods (squared vs. linear), request-dependent penalties, and varying network capacities not only underscores the complexity of designing a robust scheduling model but also identifies which approaches yield the best results.

8.2. Discussion

The discussion examines the findings of this thesis in relation to existing literature, identifying the key contributions, implications, and limitations of the proposed dynamic scheduling model. It highlights how the research addresses gaps in current literature, and situates the results within the broader context of rail freight logistics. Additionally, the section outlines practical challenges and areas for further exploration to enhance the model's applicability and effectiveness.

8.2.1. Comparison with Literature

This thesis builds upon and extends existing research in several key areas. Unlike prior studies focusing predominantly on pricing mechanisms, such as Armstrong and Meissner (2010) and Li et al. (2024), this research introduces a dynamic scheduling approach aimed at minimizing time deviations and forecasted congestion without relying on pricing strategies. This is particularly relevant in the Dutch rail freight context, where prices are fixed by governmental agreements and cannot be adjusted to influence customer behavior.

In terms of demand forecasting, the use of the deterministic demand model combined with the C-Logit framework and dynamic scheduling of requests is a novel contribution. While studies like Kraft (2002) emphasize on dynamic scheduling through demand prediction, they largely focus on car-to-train allocations or dynamic pricing strategies. By using the demand forecasting focused on the network and distributing this using the C-Logit model, the dynamic scheduling focuses more on dynamics scheduling through itinerary allocation. Additionally, stochastic methods such as Van Slyke and Young (2000) have been applied in industries like airlines and hotels, but their application to freight railways remains limited, particularly when offering alternatives to customers is considered.

The scheduling model presented in this thesis also introduces an objective function balancing time deviations and congestion, creating a trade-off which is investigated using a sensitivity analysis between these two objectives. Previous studies, such as Cacchiani and Toth (2012) and Cacchiani et al. (2010), investigate path-based scheduling but focus on static networks and pre-defined timetables. By contrast, this thesis evaluates dynamic scheduling scenarios with real-time adjustments, contributing insights into how congestion penalties influence itinerary selection.

Finally, the integration of revenue management principles without direct pricing mechanisms addresses a gap in freight train scheduling research. While Crevier et al. (2012) explore integrated models combining operations and revenue management, these approaches often prioritize revenue maximization over practical considerations like minimizing extreme deviations or addressing customer satisfaction directly. This thesis, therefore, represents a significant step toward practical, customer-centric solutions for rail freight logistics.

8.2.2. Limitations

This research encountered several limitations that affect the applicability of the findings. A significant simplification was the assumption of uniform train velocity. In reality, freight trains often operate at different speeds, with station waiting times and overtaking capabilities further influencing network dynamics. Implementing these factors would increase the model's complexity and computational demands but could also lead to a broader range of scheduling options. The increase in options could either improve or worsen the outcome of the results, depending on the scenarios tested. Incorporating these variations would make the model more reflective of real-world operations and enhance its practical relevance.

Another limitation was starting with an empty network. This assumption is unrealistic, as passenger trains and other repeated services are typically pre-scheduled. Including these pre-existing schedules in the model would provide a more accurate representation of the network's capacity and constraints, potentially altering the results.

The use of predefined paths for origin-destination pairs also constrained the model's flexibility. Although this limitation reflects the real world operations, allowing the model to generate paths dynamically increases the amount of scheduling options, which could make it easier to reduce departure time deviations. However, the extra path options could also include paths with longer travel times for some customers, which might negatively affect customer satisfaction. Balancing these trade-offs would be an important area for further exploration.

Due to data limitations, the model's capacity constraints were overly simplified. In reality, ProRail sets specific headway times for each station and arc, which vary significantly across the network. Applying a uniform capacity does not accurately reflect real-life scheduling and may create bottlenecks that ProRail has already mitigated through infrastructure expansion or operational adjustments. Incorporating more accurate, arc-specific capacity data would improve the model's realism and make its outcomes more representative for comparison with actual scheduling practices.

The model faced additional data limitations that impacted its realism. A complete list of distances between all stations was unavailable, requiring the use of default values for missing data. While incorporating an accurate distance matrix would better align the model with real-world scenarios, it is unlikely to significantly alter the outcomes. Furthermore, the assumption that all railway tracks are single-track does not reflect actual conditions, where many routes have multiple tracks. Implementing realistic track configurations would provide more accurate capacity representations between stations and could meaningfully influence the results.

The congestion factor calculation could also be more extensive, although this mostly just acts as an estimation, it now only takes the account of overlap between paths into account as part of the utility. This lacks potentially other important influences like path length or travel time. Including more elaborate utility calculations, changes the forecasted demand distribution per path which changes the congestion factor. By implementing this approach for the congestion factor, outcome results will be different, possibly improving or worsening the model.

Finally, the study relied on request data from a single day, which restricts the scope of the analysis. While network capacity could be adjusted, testing the algorithm with different request lists could reveal whether its performance depends on the specific request order used. This limitation highlights the importance of evaluating the model under diverse scenarios to better understand its robustness and applicability.

8.3. Recommendations

This section provides practical suggestions for improving ProRail's scheduling practices and outlines potential directions for future research. The recommendations focus on enhancing demand forecasting, optimizing capacity utilization, and refining scheduling models to address real-world complexities more effectively. These insights aim to guide both ProRail and researchers in the advancement of freight train scheduling methods to improve efficiency and customer satisfaction.

8.3.1. Practical Recommendations for ProRail

Based on the insights gained during this thesis, several recommendations can be made for ProRail to improve its scheduling research and practices.

ProRail could extend this research by incorporating recurring requests, such as those submitted for every Wednesday or the first day of each month, to better reflect real-world scenarios. Including recurring requests would change the scheduling approach. For example, a request for the same departure time every week should ideally be approved as a single request, rather than treated as 52 separate requests (one for each week of the year), as handling them individually would complicate the logistics for all parties involved. While this approach may reduce scheduling flexibility and alter the results, it would offer a more realistic representation of actual operations.

Developing demand forecasting tools would enable ProRail to predict where and when requests will likely be submitted. With access to extensive data and the repetitive nature of train scheduling, ProRail is well-positioned to create accurate forecasts. Researching forecasting accuracy could identify uncertainty levels, which can then be included into the model. This would result in higher variability in outcomes, but would more accurately reflect real-world implementation.

Another important consideration is researching the impact of the trade-off between average time deviations and extreme outliers. For example, a scenario where all customers face an average departure time deviation of 10 minutes with no deviations exceeding 100 minutes might be more acceptable than one where most customers experience no departure time deviations, but a few endure extreme deviations of 800 minutes. Researching stakeholder perspectives, including ProRail and its customers, could help determine what outcomes benefit customer satisfaction the most.

Additionally, ProRail should prioritize collecting data on denied requests and analyzing the reasons behind these denials. Currently, the reasons for cancellations are unclear. For example, it is unknown whether a request is canceled because the customer prefers a different departure time, wants to adjust the number of cars on the train, or if a slight schedule adjustment has been made. Without this information, it is difficult to determine whether cancellations are due to customer dissatisfaction or operational adjustments. Gathering this data would provide valuable insights into customer behavior and decision-making processes, helping to refine scheduling strategies and better align them with customer needs.

8.3.2. Future research

Future research could explore different methods for adjusting the congestion factor (CF) penalty over time. Currently, the CF decreases linearly per request, but experimenting with an exponential decrease could yield different results, allowing the penalty to dominate more strongly early on and taper off after a certain threshold. Alternatively, a simpler on/off switch could also be tested, where the CF penalty is only active for a set percentage of requests before being deactivated.

The current model shows that squaring CF values per section improves results, but further experiments could test raising the CF to higher powers, such as cubing or even higher exponents. This would widen the gap between low and high CF values, potentially influencing the allocation of itineraries in new ways. Another approach could involve squaring the CF for entire itineraries, which might favor some alternatives with slightly higher time deviations but significantly lower congestion factors.

One issue with the current model is that it allows unnecessary switches to itineraries with time deviations (which have a CF close to zero), even though the original CF for the preferred departure time was already favorable. These switches could be prevented by combining a weighted sum approach with parametric sensitivity analysis. In this case this would lead to the objective function acting more as a constraint: if an itinerary with zero time deviation has a CF above a set threshold (e.g., 1), the objective function would be activated to identify possibly better alternatives. If the CF is below the threshold, the itinerary with zero time deviation would be offered without any further optimization.

Due to not all the values for w being checked in the repeated sensitivity analysis and this being computationally time consuming, an idea for future research could be to store the best values for w resulting from individual experiments and use these values as an input for the repeated sensitivity analysis. Running the repeated sensitivity analysis with these values for w could provide more consistent improvements.

Another improvement could focus on minimizing time intervals between sections for consecutive orders. The current model allows requests to be scheduled at their preferred departure times without optimizing for tighter scheduling. For instance, if a train departs at 12:00 and the minimum interval is 10 minutes, the next train could depart at 12:10. However, if a request prefers 12:15, it is allocated at that time, leaving a gap of 5 minutes. Shifting this request to 12:10 could enhance scheduling efficiency. This approach could also extend beyond just the departure time of the entire itinerary, by considering the departure time of each section, adjusting schedules by a few minutes to pack itineraries and their sections closer together, improving tighter scheduling in the network.

Future research should refine the calculation of remaining arc capacity per hour leading to a greater accuracy in the congestion factor calculation. The current approach, which subtracts the number of scheduled trains from the maximum capacity for an arc, does not account for inefficient scheduling. In practice, an arc may be fully scheduled at only 60–80% of its theoretical capacity due to suboptimal train spacing. Developing a capacity model that evaluates how many trains can still be scheduled within an hour, considering consecutive arc availability and headway times, could enhance the reliability of congestion factor estimates, leading to better informed scheduling decisions.

Future research should also refine the calculation of remaining arc capacity to improve accuracy. The current method, of subtracting the amount of scheduled trains on an arc from its maximum capacity, does not portray correct capacity management. If requests are scheduled inefficiently, it could be that an arc can only fit 80% or 60% of the maximum capacity before being fully scheduled. Developing a capacity model which checks how many trains can still be scheduled on an arc in an hour, by checking the amount of consecutive arcs in time being available also accounting for the headway time, could enhance the reliability of congestion estimates, leading to better informed scheduling decisions and more efficient use of network resources.

Demand prediction accuracy is another critical area for investigation. While this study assumes deterministic demand, real-world predictions are never 100% accurate. Future research should evaluate how varying levels of prediction accuracy (e.g., 90% or 80%) impact the model's outcomes. Additionally, studies could explore the practical limits of forecasting accuracy on the prediction of train requests.

Experiments could investigate how often specific values of w improve the scheduling model's performance. Collecting data on optimal weights across scenarios would allow future research to focus on targeted experiments rather than the limited weights tested in this thesis. Since the best value for w differs per request list, dynamically adjusting w based on the request list would be beneficial. To refine this process, an AI model could be trained to predict the optimal w based on the initial request list. This approach could enable dynamic adjustments, increasing the likelihood of better results.

Another topic for future research is establishing a robust link between the algorithm's outcomes on departure time deviation per request and customer satisfaction. Kraft (2002) provides a formula to calculate acceptance probabilities based on departure time deviations, which can serve as a foundational model for understanding the impact of time deviations on customer acceptance. The formula is given as:

$$P(\text{Accepts Quote}) = \frac{e^{\alpha-\beta\Delta}}{1+e^{\alpha-\beta\Delta}}$$

where Δ represents the departure time deviation, and α and β are calibration parameters. For example, Kraft (2002) assumes that a departure time deviation of two days corresponds to a 50% acceptance probability, using empirical estimates. Substituting $\rho = 2$, $\alpha = 4.394$, and $\beta = \alpha/\rho$, the formula simplifies to:

$$P(\text{Accepts Quote}) = \frac{e^{4.394-4.394/\rho}}{1+e^{4.394-4.394/\rho}}$$

This formula is based on several assumptions and serves as an estimate rather than a definitive measure. While it provides a useful starting point, tailoring this model to the Dutch rail freight context would require ProRail to conduct research and calibrate these parameters based on local customer behavior.

In addition to time deviations, other factors likely influence customer satisfaction, but are not captured in Kraft's model. These include the lead time of the request (e.g., requests submitted two weeks versus one year in advance), whether the deviation shifts departure from day to night, the frequency of rescheduling for recurring requests (e.g., multiple small shifts versus one large shift), the trip length, the type of freight transported, etc. Research incorporating these variables, supported by empirical data on customer acceptance collected by ProRail, could significantly enhance understanding of customer behavior. Using AI models trained on this dataset could further improve predictions of customer satisfaction, enabling more tailored and effective scheduling decisions.

References

Alaghband, M., & Farhang Moghadam, B. (2019, December). *An Optimization Model for Scheduling Freight Trains on a Single Rail Track*. (Cit. on p. 13).

Armstrong, A., & Meissner, J. (2010). Railway Revenue Management: Overview and Models. *Lancaster University Management School* (cit. on pp. 3, 4, 13, 73).

Ben-Akiva, M., & Bierlaire, M. (1999). Discrete Choice Methods and their Applications to Short Term Travel Decisions. In R. W. Hall (Ed.), *Handbook of Transportation Science* (pp. 5–33). Springer US. https://doi.org/10.1007/978-1-4615-5203-1_2 (cit. on pp. 23, 24).

Brännlund, U., Lindberg, P., Nõu, A., & Nilsson, J.-E. (1998). Railway timetabling using Lagrangian relaxation. *Transportation Science*, *32*(4), 358–369. https://doi.org/10.1287/trsc.32.4.358 (cit. on p. 12).

Cacchiani, V., Caprara, A., & Toth, P. (2010). Scheduling extra freight trains on railway networks. *Transportation Research Part B: Methodological*, *44*(2), 215–231. https://doi.org/10.1016/j.trb.2009.07.007 (cit. on pp. 3, 4, 15, 19, 73).

Cacchiani, V., & Toth, P. (2012). Nominal and robust train timetabling problems. *European Journal of Operational Research*, *219*(3), 727–737. https://doi.org/10.1016/j.ejor.2011.11.003 (cit. on pp. 3, 4, 15, 73).

Caprara, A., Monaci, M., Toth, P., & Guida, P. (2006). A Lagrangian heuristic algorithm for a real-world train timetabling problem. *Discrete Applied Mathematics*, *154*(5 SPEC. ISS.), 738–753. https://doi.org/10.1016/j.dam.2005.05.026 (cit. on p. 12).

Caprara, A., Kroon, L., Monaci, M., Peeters, M., & Toth, P. (2007, January). Chapter 3 Passenger Railway Optimization. In C. Barnhart & G. Laporte (Eds.), *Handbooks in Operations Research and Management Science* (pp. 129–187, Vol. 14). Elsevier. https://doi.org/10.1016/S0927-0507(06)14003-7 (cit. on p. 12).

Cascetta, E., Nuzzolo, A., Russo, F., & Vitetta, A. (1996). A MODIFIED LOGIT ROUTE CHOICE MODEL OVERCOMING PATH OVERLAPPING PROBLEMS. SPECIFICATION AND SOME CALIBRATION RESULTS FOR INTERURBAN NETWORKS. Retrieved October 30, 2024, from https://www.semanticscholar.org/paper/A-MODIFIED-LOGIT-ROUTE-CHOICE-MODEL-OVERCOMING-PATH-Cascetta-Nuzzolo/501eecf9311a9e60ae8be1f668319ca11fafc6a3 (cit. on pp. 13, 15, 24).

Centraal Bureau voor de Statistiek. (2024). Vrachtvervoer over het spoor [Accessed: 2024-06-04]. https://www.cbs.nl/nl-nl/visualisaties/verkeer-en-vervoer/goederen/spoor/vracht (cit. on p. 1).

Compendium voor de Leefomgeving. (2018). Aanbod openbaar vervoer, 2000-2017 [Accessed: 2024-11-12]. https://www.clo.nl/indicatoren/nl214004-aanbod-openbaar-vervoer-2000-2017 (cit. on p. 42).

Crainic, T., Ferland, J.-A., & Rousseau, J.-M. (1984). TACTICAL PLANNING MODEL FOR RAIL FREIGHT TRANSPORTATION. *Transportation Science*, *18*(2), 165–184. https://doi.org/10.1287/trsc.18.2.165 (cit. on p. 12).

Crevier, B., Cordeau, J.-F., & Savard, G. (2012). Integrated operations planning and revenue management for rail freight transportation. *Transportation Research Part B: Methodological*, *46*(1), 100–119. https://doi.org/10.1016/j.trb.2011.09.002 (cit. on pp. 3, 4, 13, 14, 74).

Davis, M. H. A., Dempster, M. A. H., Sethi, S. P., & Vermes, D. (1987). Optimal Capacity Expansion under Uncertainty [Publisher: Applied Probability Trust]. *Advances in Applied Probability*, *19*(1), 156–176. https://doi.org/10.2307/1427378 (cit. on pp. 13, 14).

European Environment Agency (EEA). (2023). *Specific co2 emissions per tonne-kilometer of freight transport by mode* [Accessed: 2023-06-05]. https://www.eea.europa.eu/data-and-maps/daviz/specific-co2-emissions-per-tonne-2#tab-chart_1 (cit. on p. 1).

Feng, F.-L., Zhang, J.-Q., & Guo, X.-F. (2015). A dynamic model for railway freight overbooking. *Journal of Central South University*, *22*(8), 3257–3264. https://doi.org/10.1007/s11771-015-2864-4 (cit. on pp. 13, 15).

Gallego, G., & Topaloglu, H. (2019). *Revenue Management and Pricing Analytics* (Vol. 279). Springer New York. https://doi.org/10.1007/978-1-4939-9606-3 (cit. on pp. 3, 13).

Ghoseiri, K., Szidarovszky, F., & Asgharpour, M. (2004). A multi-objective train scheduling model and solution. *Transportation Research Part B: Methodological*, *38*(10), 927–952. https://doi.org/10.1016/j.trb.2004.02.004 (cit. on pp. 12, 15).

Gurobi Optimization, L. (2024). *Gurobi optimizer reference manual*. https://www.gurobi.com (cit. on p. 39).

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., … Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, *585*(7825), 357–362. https://doi.org/10.1038/s41586-020-2649-2 (cit. on p. 39).

Hunter, J. D. (2007). Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, *9*(3), 90–95. https://doi.org/10.1109/MCSE.2007.55 (cit. on p. 40).

Inc., P. T. (2024). Plotly: Modern data visualization. https://plotly.com (cit. on p. 40).

Infrasite. (2023). *Aanvraag en verdeling spoorcapaciteit* [Accessed: 2023-06-05]. https://www.infrasite.nl/glossary/aanvraag-en-verdeling-spoorcapaciteit/?gdpr=deny (cit. on p. 1).

Janssen, S. (2023, December). Model contract of use 2024 - 6 december 2023 [Available online: https://www.prorail.nl/siteassets/homepage/samenwerken/vervoerders/documenten/netwerkverklaring-2025/model-contract-of-use-2024-version-1.1---2-february-2024.pdf]. (Cit. on p. 4).

Jovanovic, D., & Harker, P. T. (1991). Tactical scheduling of rail operations. The SCAN I system. *Transportation Science*, *25*(1), 46–64. https://doi.org/10.1287/trsc.25.1.46 (cit. on pp. 13, 15).

Kraft, E. (2002). Scheduling railway freight delivery appointments using a bid price approach. *Transportation Research Part A: Policy and Practice*, *36*(2), 145–165. https://doi.org/10.1016/S0965-8564(00)00041-0 (cit. on pp. 2–4, 13, 15, 16, 73, 76).

Li, D., Islam, D. M. Z., Robinson, M., Song, D.-P., Dong, J.-X., & Reimann, M. (2024). Network revenue management game in the railway industry: Stackelberg equilibrium, global optimality, and mechanism design. *European Journal of Operational Research*, *312*(1), 240–254. https://doi.org/10.1016/j.ejor.2023.06.044 (cit. on pp. 3, 4, 13, 73).

Lucía Morales. (2020). *World gdp growth: Annual changes by global contribution* [Accessed: 2023-06-05]. https://www.researchgate.net/figure/World-GDP-Growth-Annual-Changes-by-Global-Contribution_fig1_369691898 (cit. on p. 1).

Office of Rail and Road (ORR). (2023). *Rail emissions statistics* [Accessed: 2023-06-05]. https://dataportal.orr.gov.uk/statistics/infrastructure-and-emissions/rail-emissions/ (cit. on p. 1).

OVNet. (2023). Spoorbegrip: Capaciteit (railway term: Capacity) [Accessed: 2024-11-12]. https://www.ovnet.nl/?spoorbegrip=capaciteit (cit. on p. 42).

pandas development team, T. (2020, February). *Pandas-dev/pandas: Pandas* (Version latest). Zenodo. https://doi.org/10.5281/zenodo.3509134 (cit. on p. 39).

ProRail. (2020). *Beheerplan 2020-2021* [Accessed: 2023-06-05]. https://www.prorail.nl/siteassets/homepage/over-ons/documenten/prorail_beheerplan_2020-2021.pdf (cit. on pp. 1, 2).

ProRail. (2021, June). *Integrale mobiliteitsanalyse 2021 - deelrapportage spoor en btm* (tech. rep.) (Accessed: 2024-06-04). ProRail. https://zoek.officielebekendmakingen.nl/blg-990633.pdf (cit. on p. 1).

ProRail. (2024a). *Grote puzzel, kleine aanpassingen* [Accessed: 2024-10-09]. https://www.prorail.nl/nieuws/grote-puzzel-kleine-aanpassingen (cit. on pp. 2, 8).

ProRail. (2024b). *Veel gedaan, nog meer te doen: Lijst met werkzaamheden* [Accessed: 2024-09-02]. https://www.prorail.nl/verhalen/veel-gedaan-nog-meer-te-doen/lijst-met-werkzaamheden (cit. on p. 40).

ProRail. (2025). Spoorkaart [Accessed: 2025-01-04]. https://www.prorail.nl/reizen/spoorkaart (cit. on pp. 44, 49).

ProRail. (n.d.). Hoe hard rijden treinen? [Accessed: 2024-11-12]. https://www.prorail.nl/veelgestelde-vragen/hoe-hard-rijden-treinen (cit. on p. 42).

Raybaut, P. (2009). Spyder-documentation. *Available online at: pythonhosted. org* (cit. on p. 39).

Rijdende Treinen. (2022). Tariefafstanden (tariff distances) [Accessed: 2024-11-12]. https://www.rijdendetreinen.nl/open-data/tariefafstanden#description (cit. on pp. 41, 42).

RTV Dordrecht. (2024). Zo veel treinen rijden er dagelijks over de betuweroute [Accessed: 2024-11-22]. https://www.rtvdordrecht.nl/nieuws/zo-veel-treinen-rijden-er-dagelijks-over-de-betuweroute#:~:text=In%20het%20topjaar%202022%20reden,een%20trein%20per%2022%20minuten. (cit. on p. 42).

Spilt, N. (n.d.). Verkortingen tabel [Accessed: 2024-11-12]. https://www.nicospilt.com/verkortingen_tabel.htm (cit. on pp. 40, 44).

Statista. (2023). Estimated worldwide real gdp growth [Accessed: 2023-06-05]. https://www.statista.com/chart/20078/estimated-worldwide-real-gdp-growth/ (cit. on p. 1).

Talluri, K. T., & Ryzin, G. J. v. (2006, February). *The Theory and Practice of Revenue Management*. Springer Science & Business Media. (Cit. on pp. 3, 13, 14).

Treinreiziger. (2020). Prorail: Meer treinen mogelijk door nieuwe planning-software [Accessed: 2024-11-12]. https://www.treinreiziger.nl/prorail-meer-treinen-mogelijk-door-nieuwe-planning-software/ (cit. on p. 42).

Van Slyke, R., & Young, Y. (2000). Finite horizon stochastic knapsacks with applications to yield management. *Operations Research*, *48*(1), 155–172. https://doi.org/10.1287/opre.48.1.155.12457 (cit. on pp. 3, 4, 15, 73).

Yan, F., & Goverde, R. M. P. (2019). Combined line planning and train timetabling for strongly heterogeneous railway lines with direct connections. *Transportation Research Part B: Methodological*, *127*, 20–46. https://doi.org/10.1016/j.trb.2019.06.010 (cit. on p. 12).

Yue, Y., Yan, X., Wang, S., Li, M., & Faghri, A. (2024). New Heuristic Algorithm Based on the Lagrangian Relaxation for a Real-World High-Speed Railway Timetabling Problem. *Transportation Research Record*, *2678*(3), 502–516. https://doi.org/10.1177/03611981231182150 (cit. on p. 12).

A
Box Plots Repeated Sensitivity Analysis

A.1. Congestion Factor Sections Squared and Scaling Congestion factor down per Request

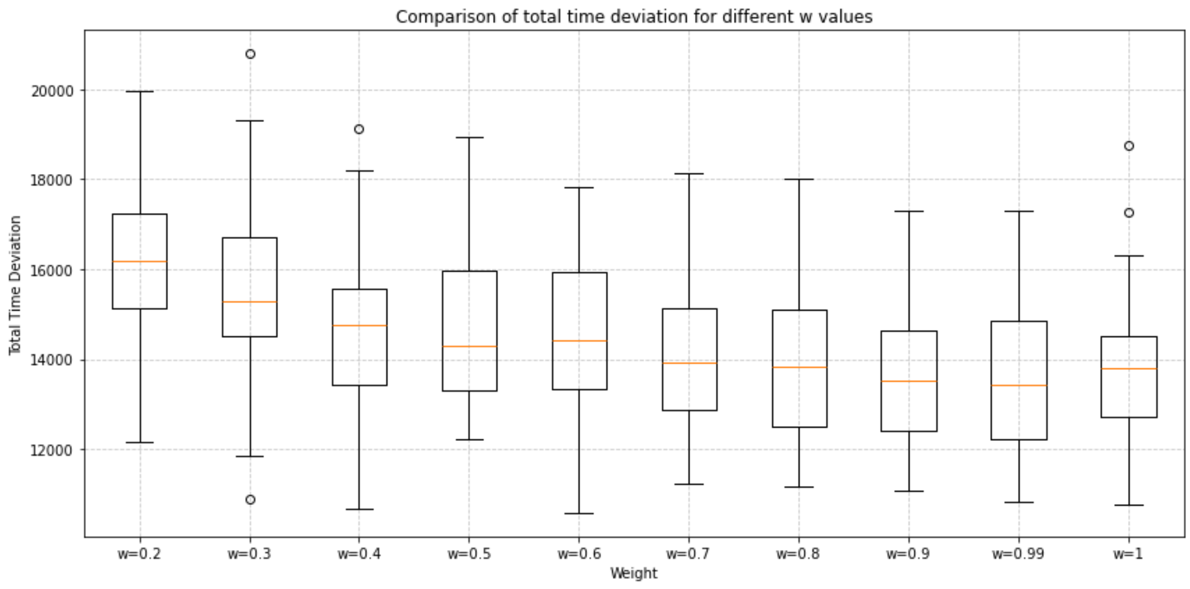


Figure A.1: Capacity = 5 trains per hour, Squared Congestion Factor, Congestion Factor Request Dependent

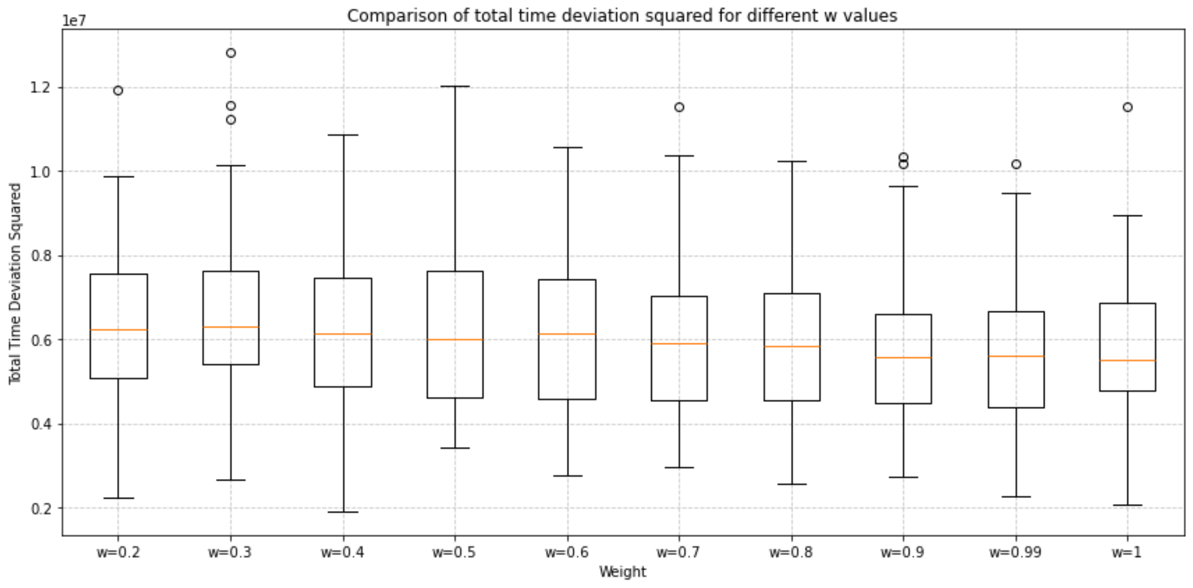


Figure A.2: Capacity = 5 trains per hour, Squared Congestion Factor, Congestion Factor Request Dependent, Total Time Deviation Squared

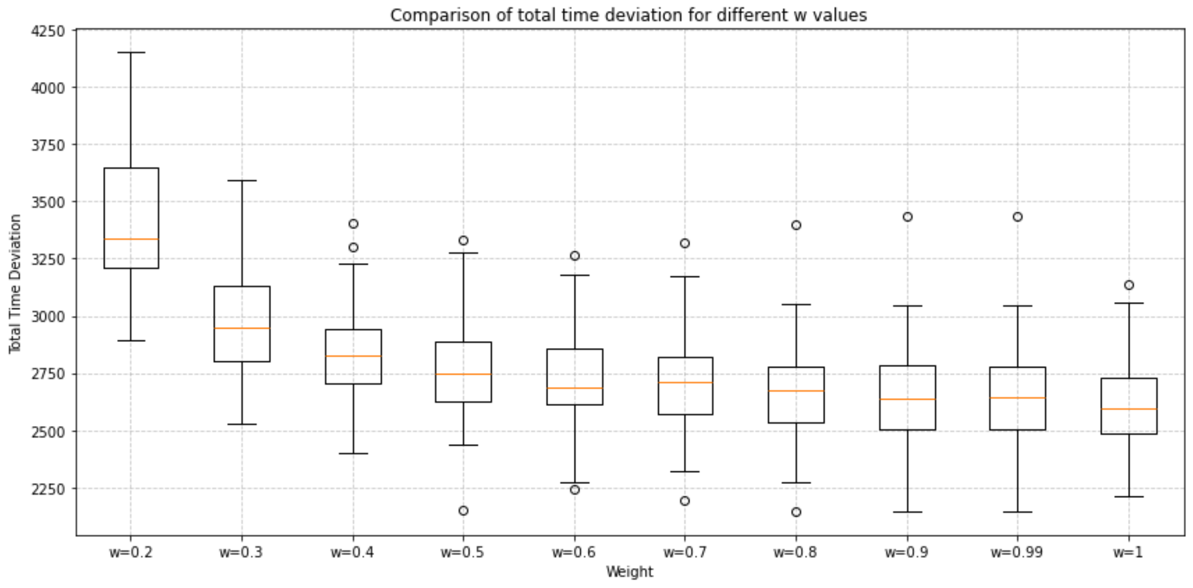


Figure A.3: Capacity = 7.5 trains per hour, Squared Congestion Factor, Congestion Factor Request Dependent

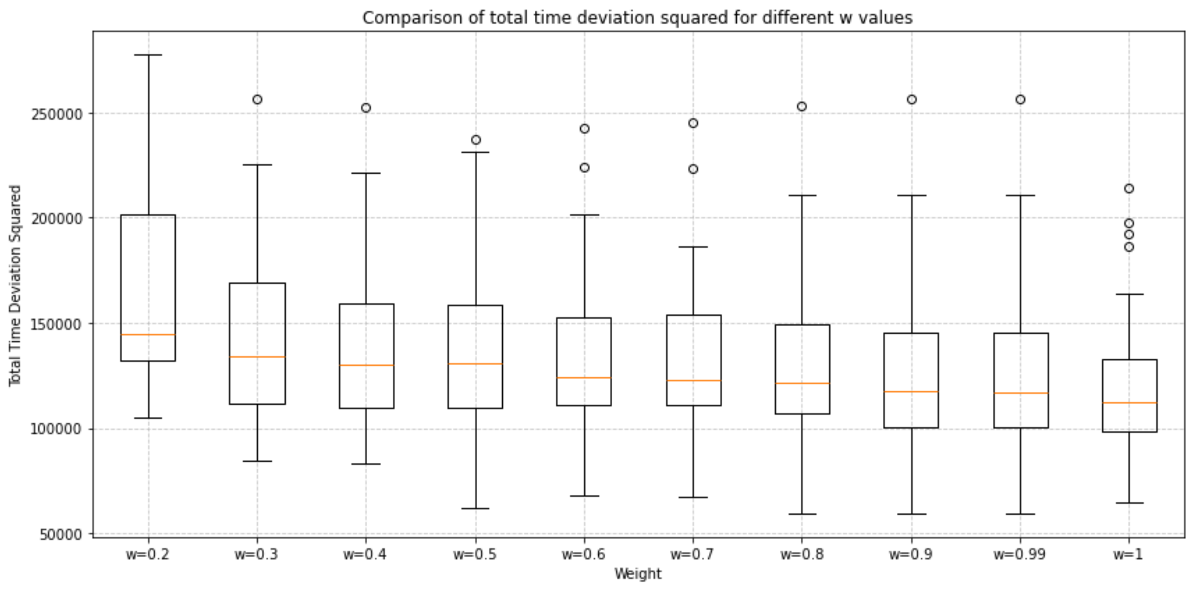


Figure A.4: Capacity = 7.5 trains per hour, Squared Congestion Factor, Congestion Factor Request Dependent, Total Time Deviation Squared

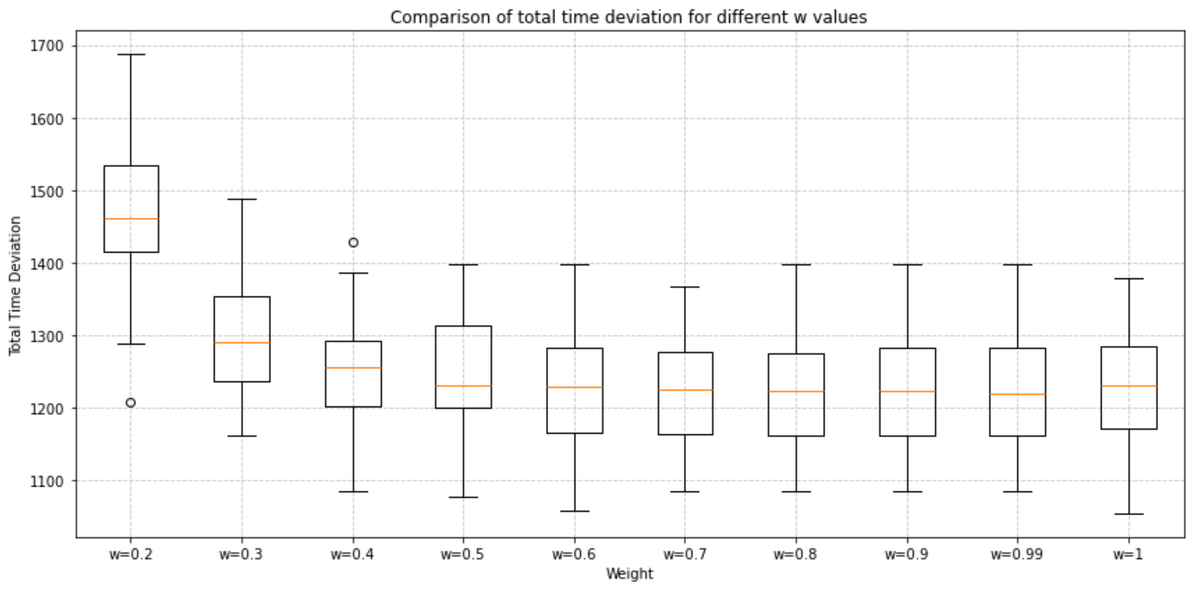


Figure A.5: Capacity = 10 trains per hour, Squared Congestion Factor, Congestion Factor Request Dependent

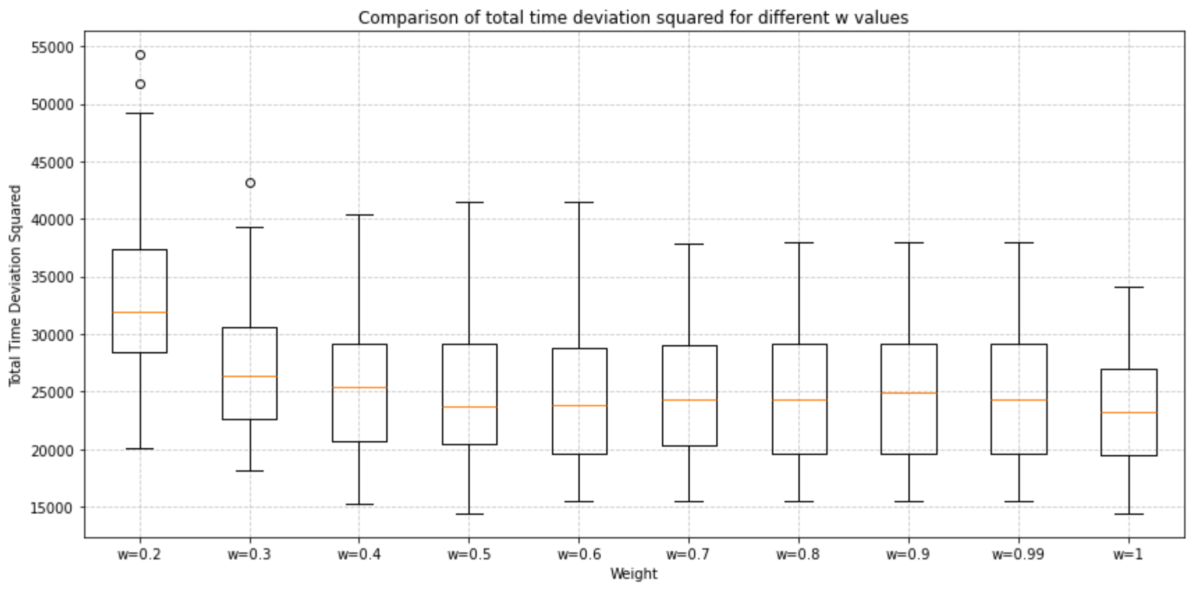


Figure A.6: Capacity = 10 trains per hour, Squared Congestion Factor, Congestion Factor Request Dependent, Total Time Deviation Squared

A.2. Congestion Factor Sections Linear and Scaling Congestion factor down per Request

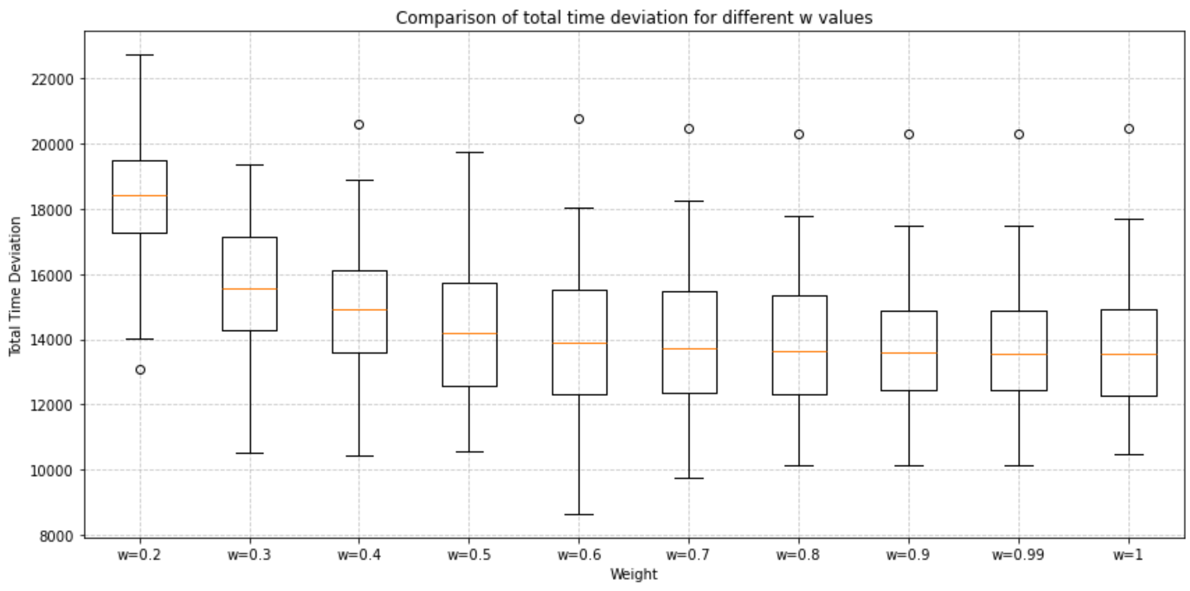


Figure A.7: Capacity = 5 trains per hour, Linear Congestion Factor, Congestion Factor Request Dependent

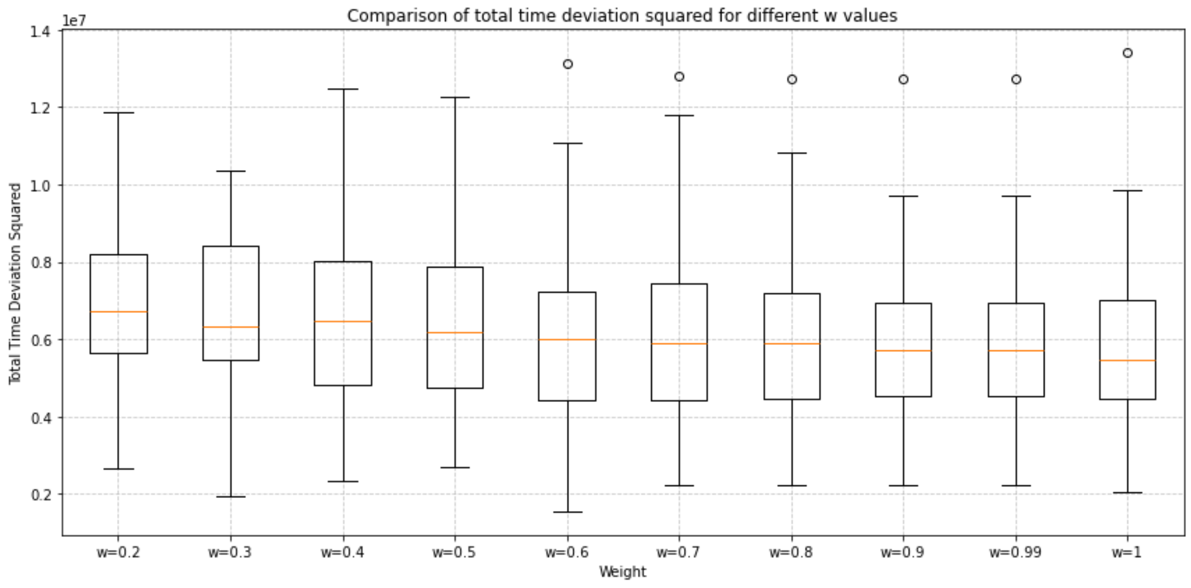


Figure A.8: Capacity = 5 trains per hour, Linear Congestion Factor, Congestion Factor Request Dependent, Total Time Deviation Squared

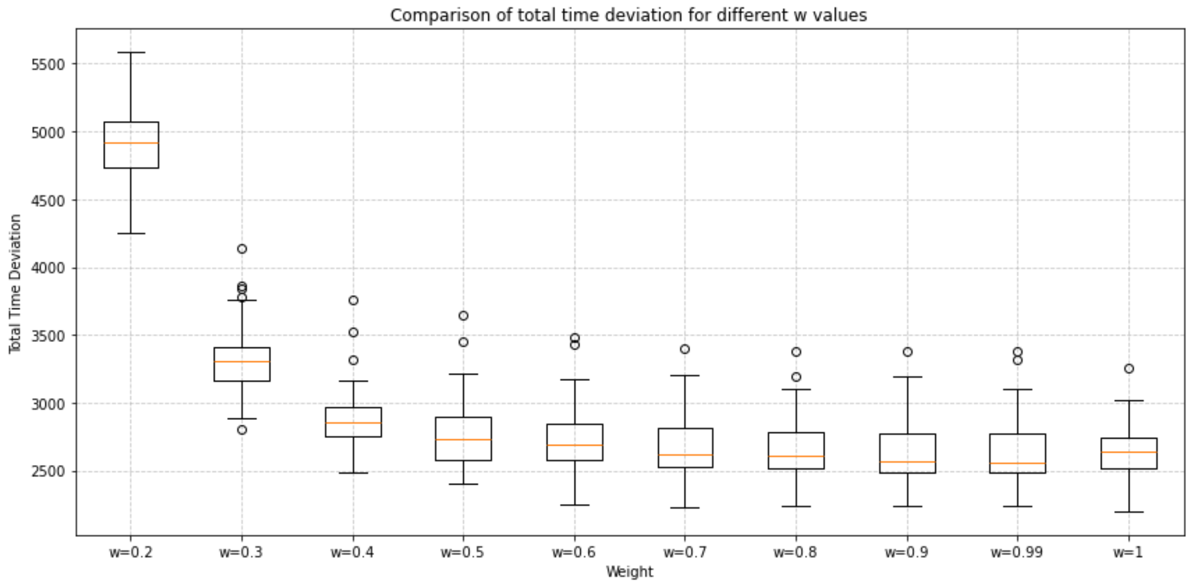


Figure A.9: Capacity = 7.5 trains per hour, Linear Congestion Factor, Congestion Factor Request Dependent

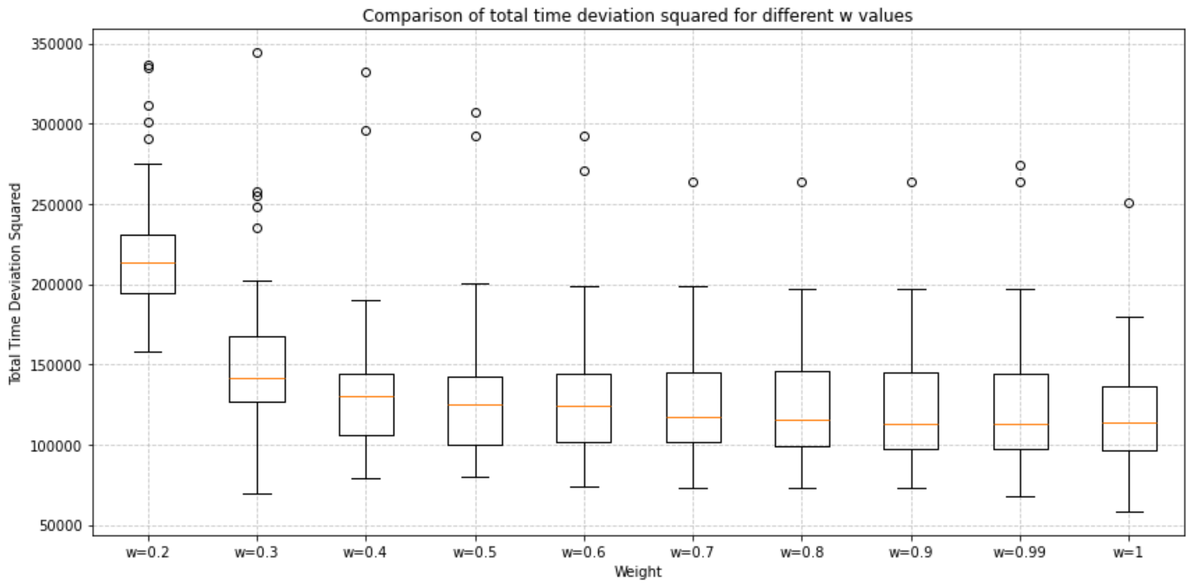


Figure A.10: Capacity = 7.5 trains per hour, Linear Congestion Factor, Congestion Factor Request Dependent, Total Time Deviation Squared

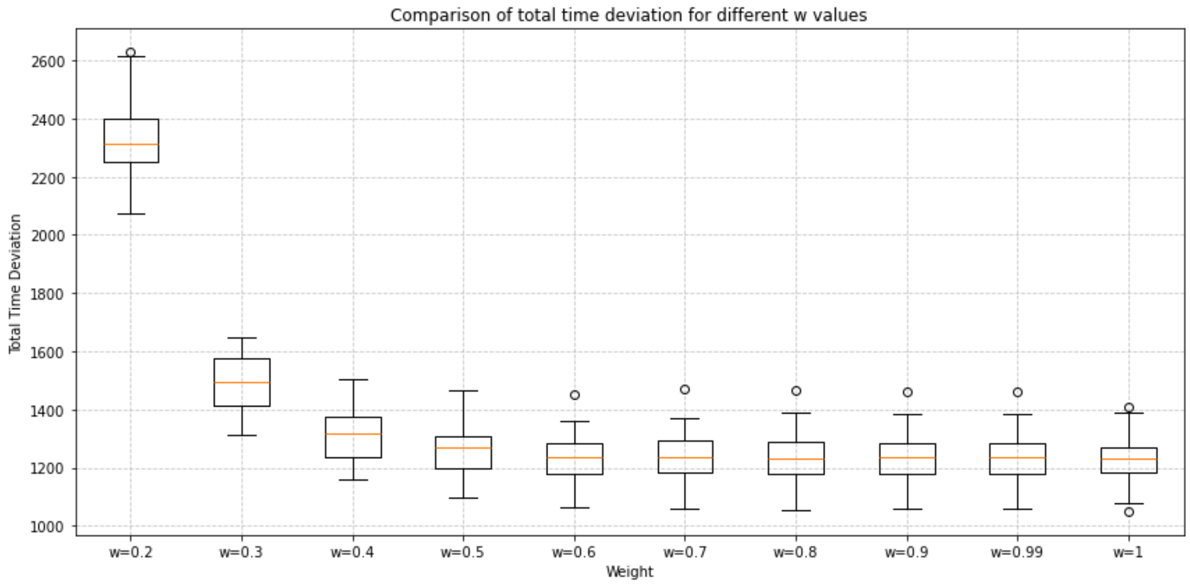


Figure A.11: Capacity = 10 trains per hour, Linear Congestion Factor, Congestion Factor Request Dependent

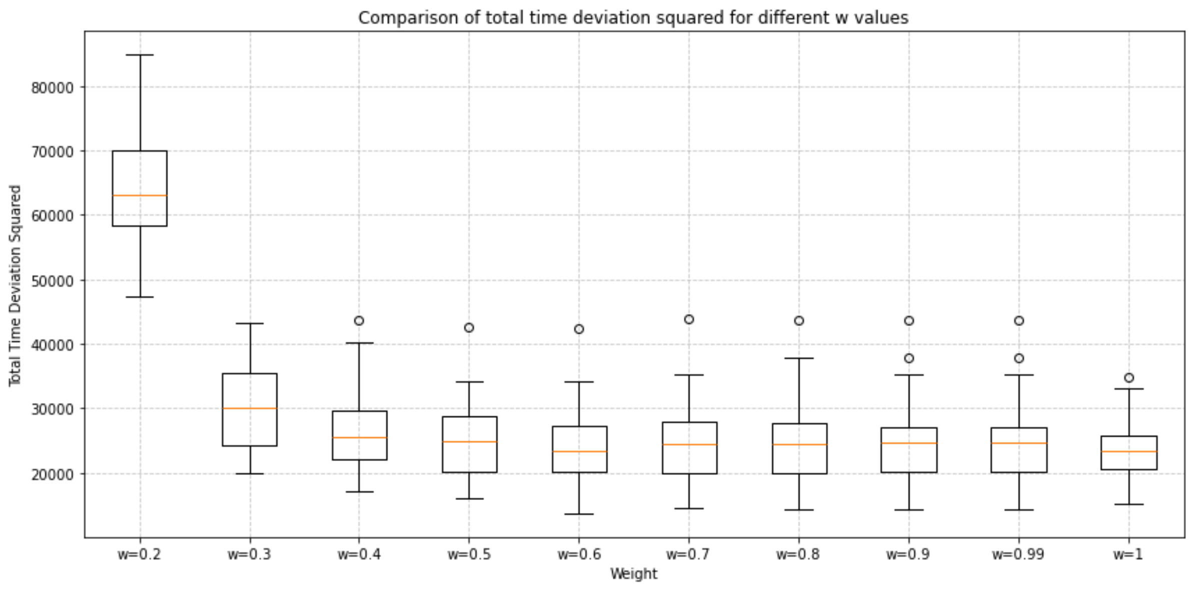


Figure A.12: Capacity = 10 trains per hour, Linear Congestion Factor, Congestion Factor Request Dependent, Total Time Deviation Squared

A.3. Congestion Factor Sections Squared and Congestion Factor Not Request Dependent

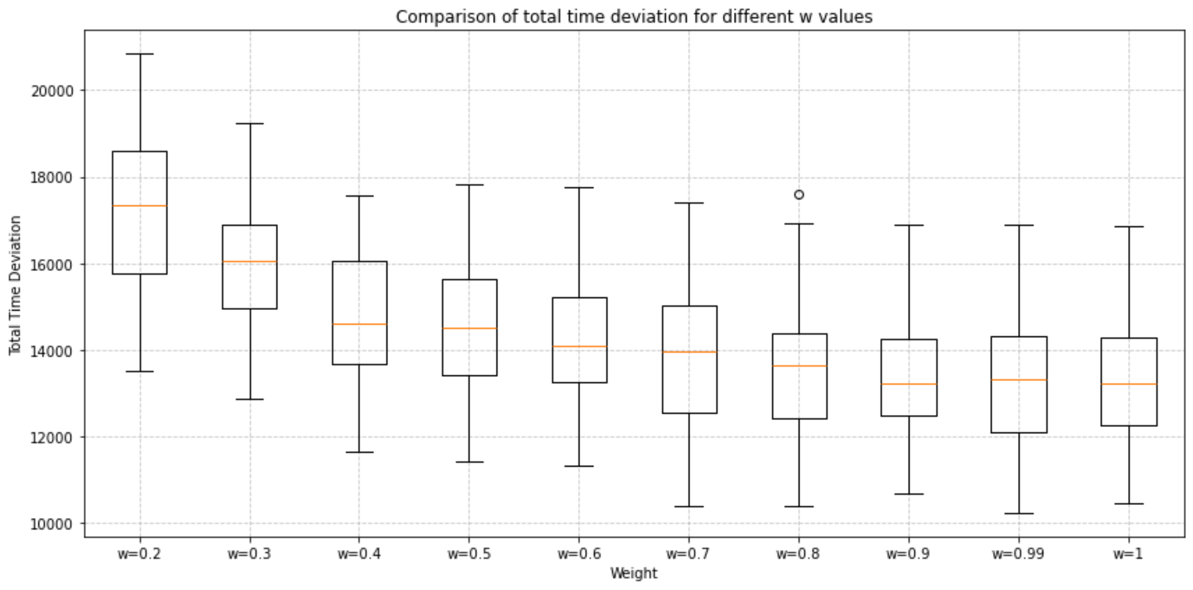


Figure A.13: Capacity = 5 trains per hour, Squared Congestion Factor, Congestion Factor Not Request Dependent

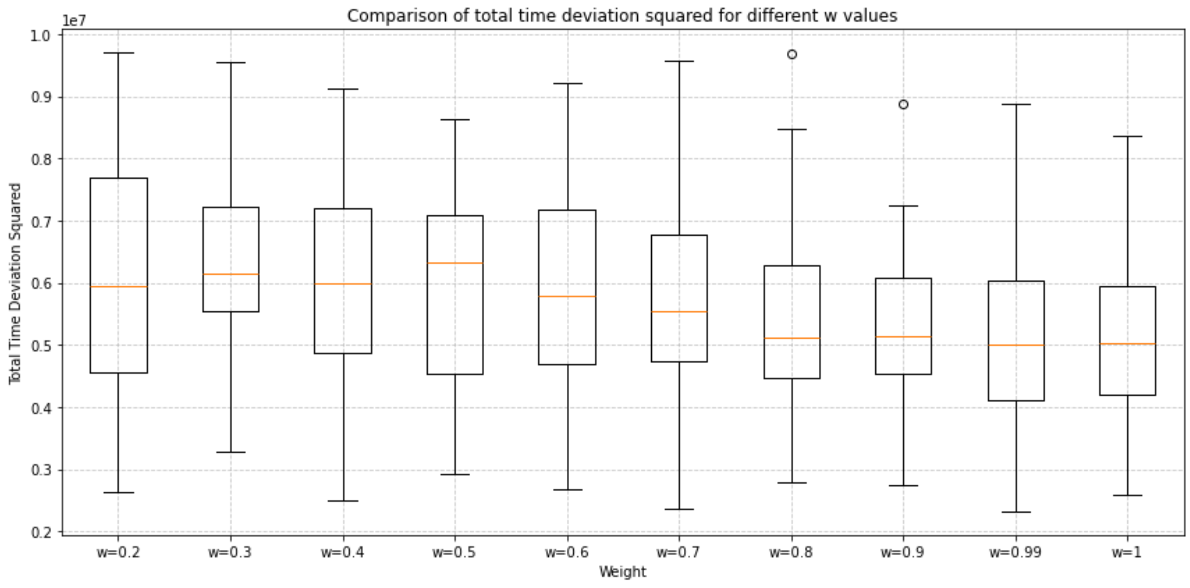


Figure A.14: Capacity = 5 trains per hour, Squared Congestion Factor, Congestion Factor Not Request Dependent, Total Time Deviation Squared

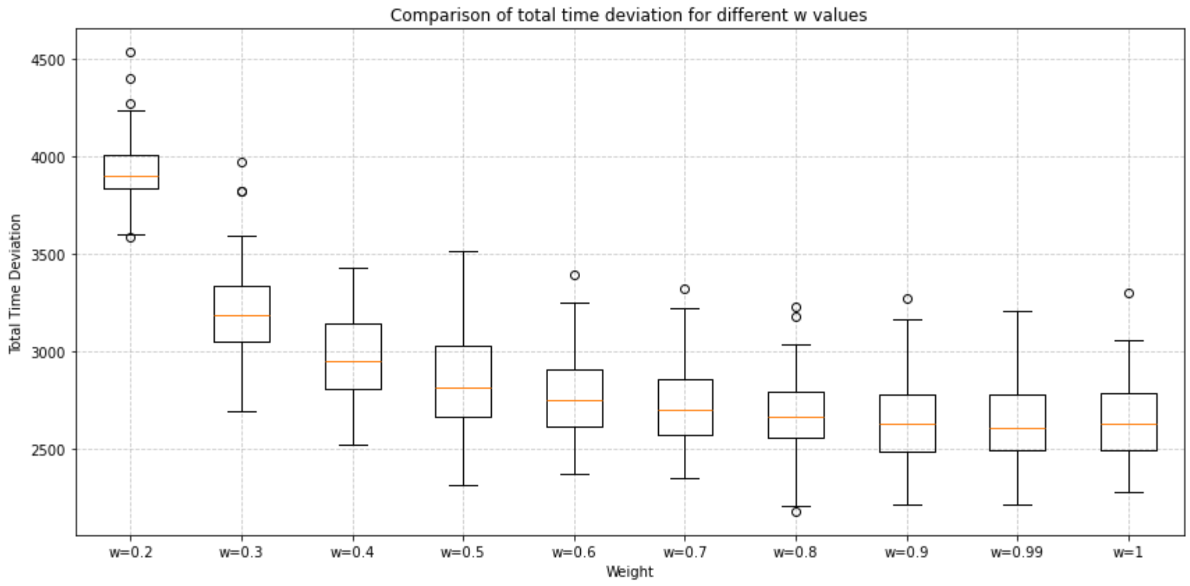


Figure A.15: Capacity = 7.5 trains per hour, Squared Congestion Factor, Congestion Factor Not Request Dependent

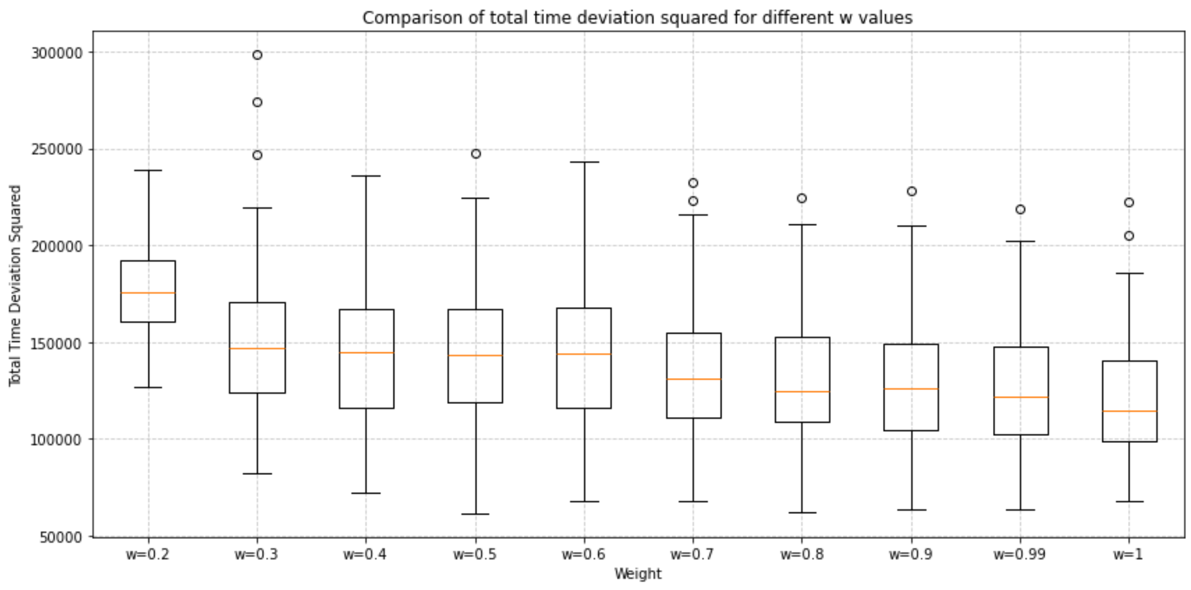


Figure A.16: Capacity = 7.5 trains per hour, Squared Congestion Factor, Congestion Factor Not Request Dependent, Total Time Deviation Squared

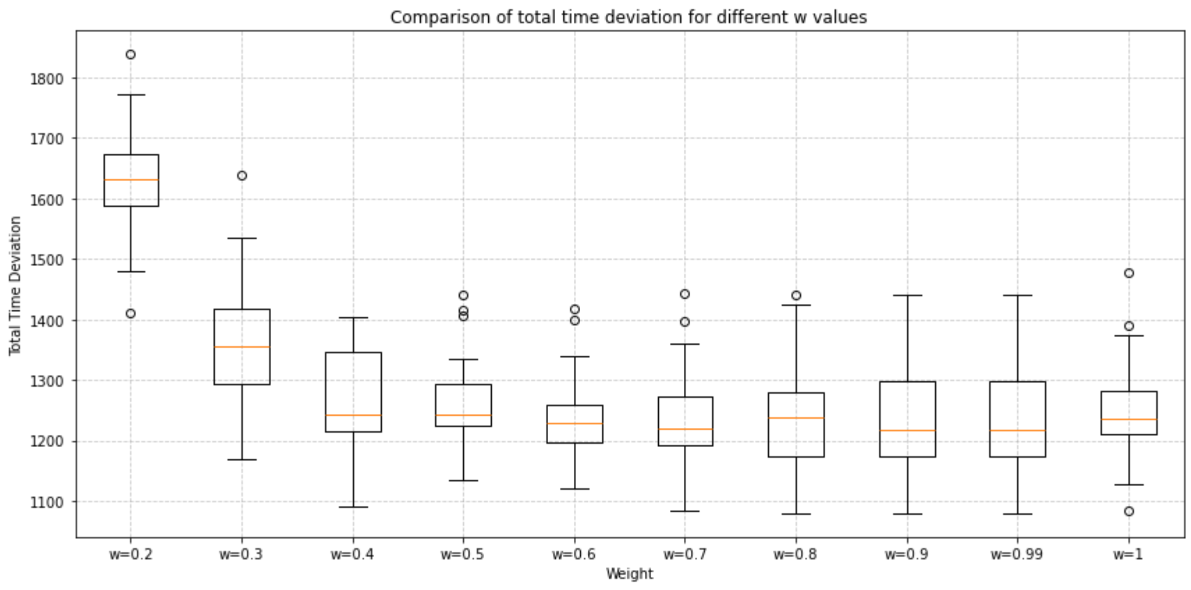


Figure A.17: Capacity = 10 trains per hour, Squared Congestion Factor, Congestion Factor Not Request Dependent

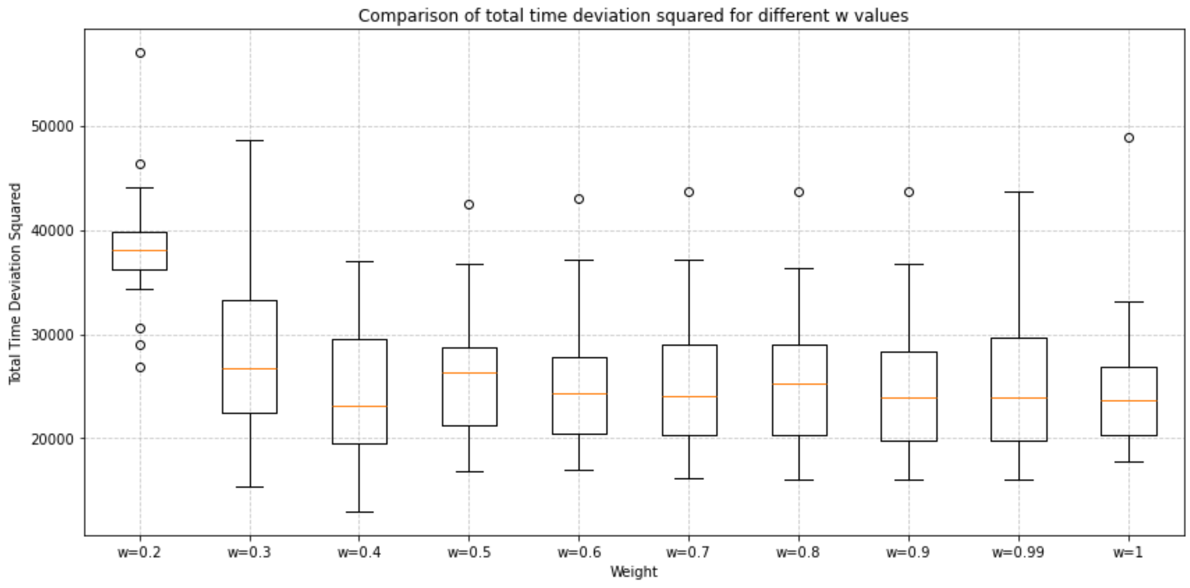


Figure A.18: Capacity = 10 trains per hour, Squared Congestion Factor, Congestion Factor Not Request Dependent, Total Time Deviation Squared

A.4. Congestion Factor Sections Linear and Congestion Factor Not Request Dependent

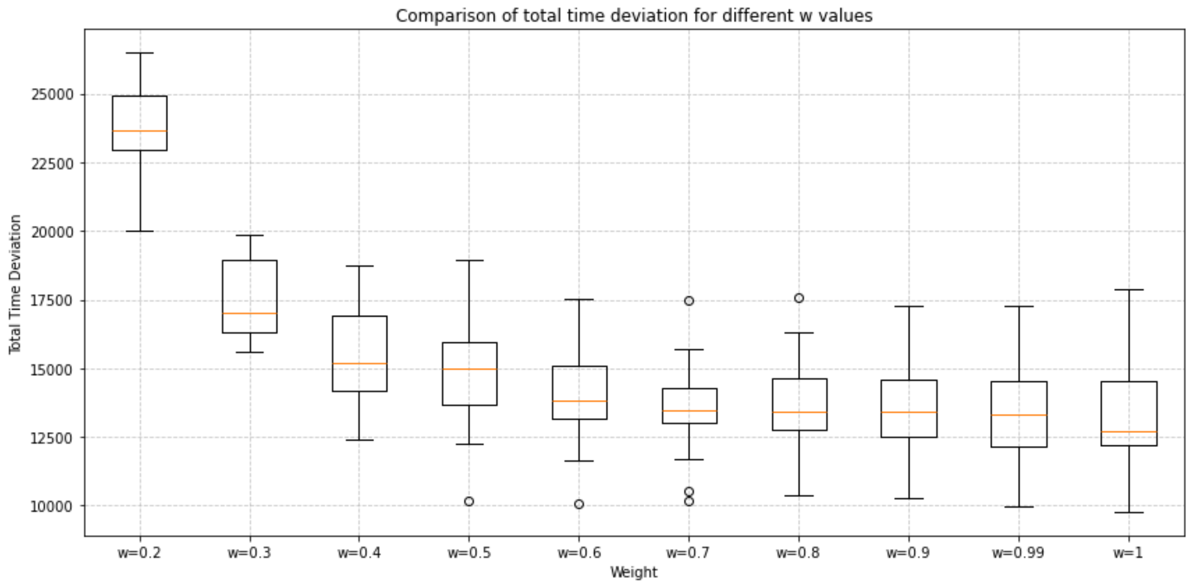


Figure A.19: Capacity = 5 trains per hour, Linear Congestion Factor, Congestion Factor Not Request Dependent

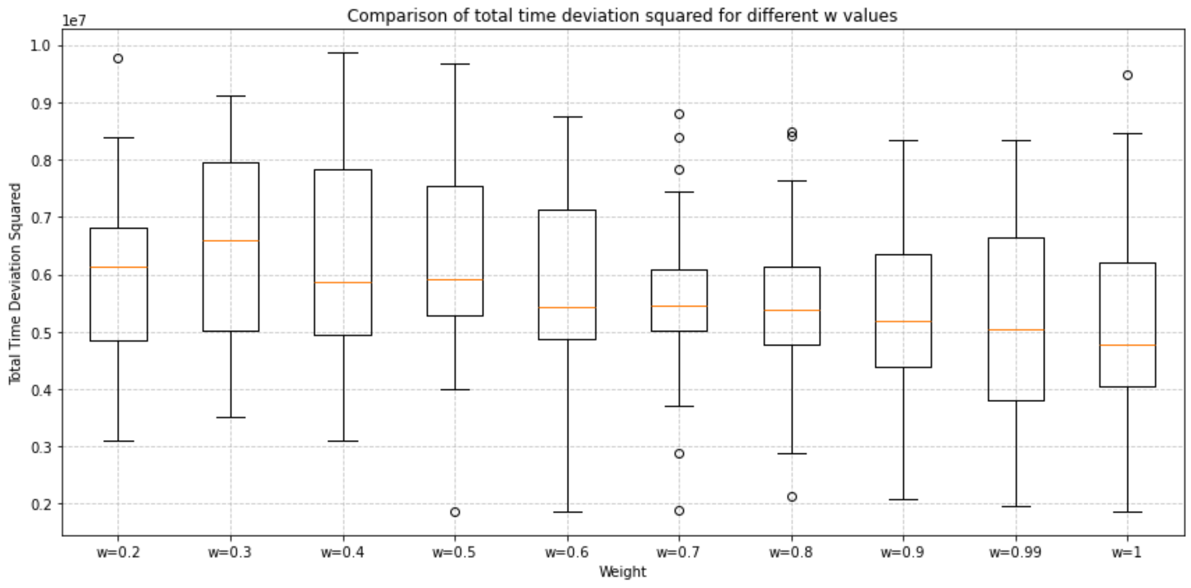


Figure A.20: Capacity = 5 trains per hour, Linear Congestion Factor, Congestion Factor Not Request Dependent, Total Time Deviation Squared

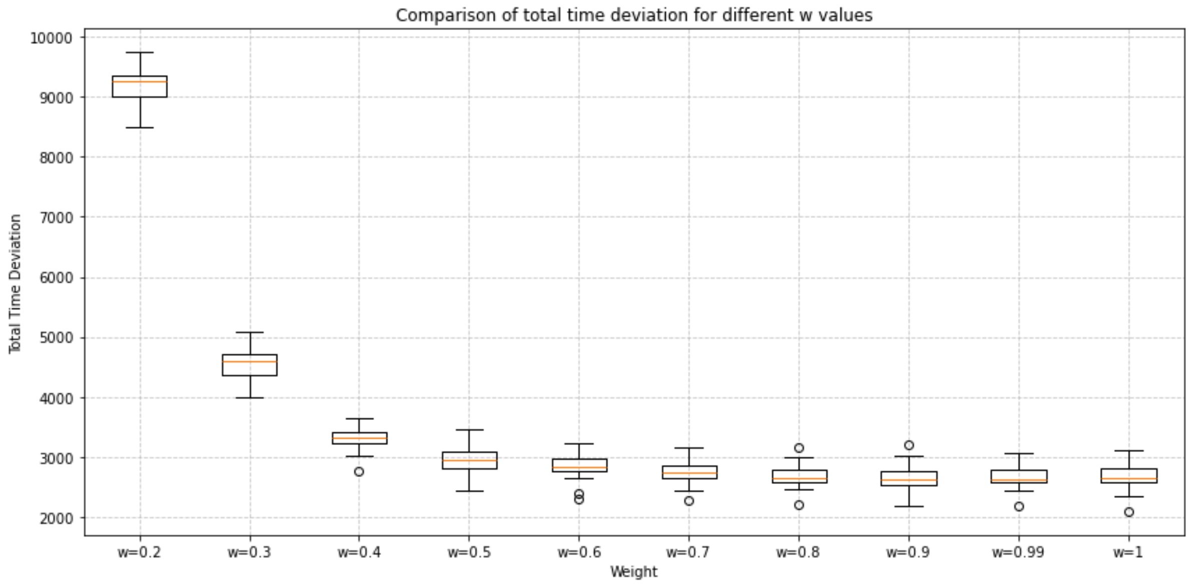


Figure A.21: Capacity = 7.5 trains per hour, Linear Congestion Factor, Congestion Factor Not Request Dependent

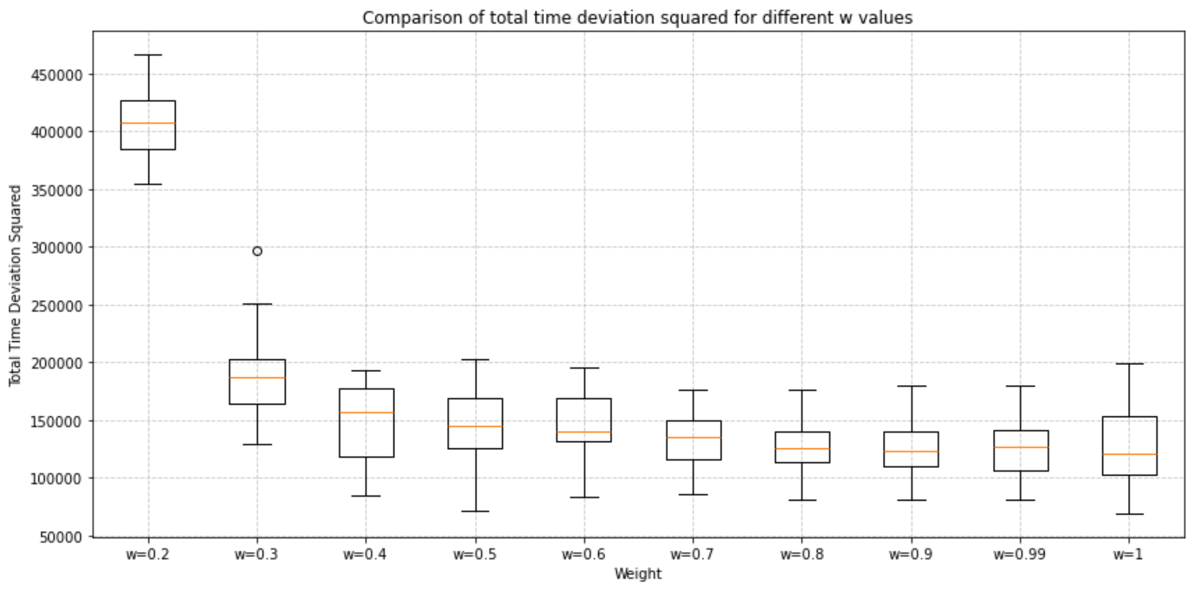


Figure A.22: Capacity = 7.5 trains per hour, Linear Congestion Factor, Congestion Factor Not Request Dependent, Total Time Deviation Squared

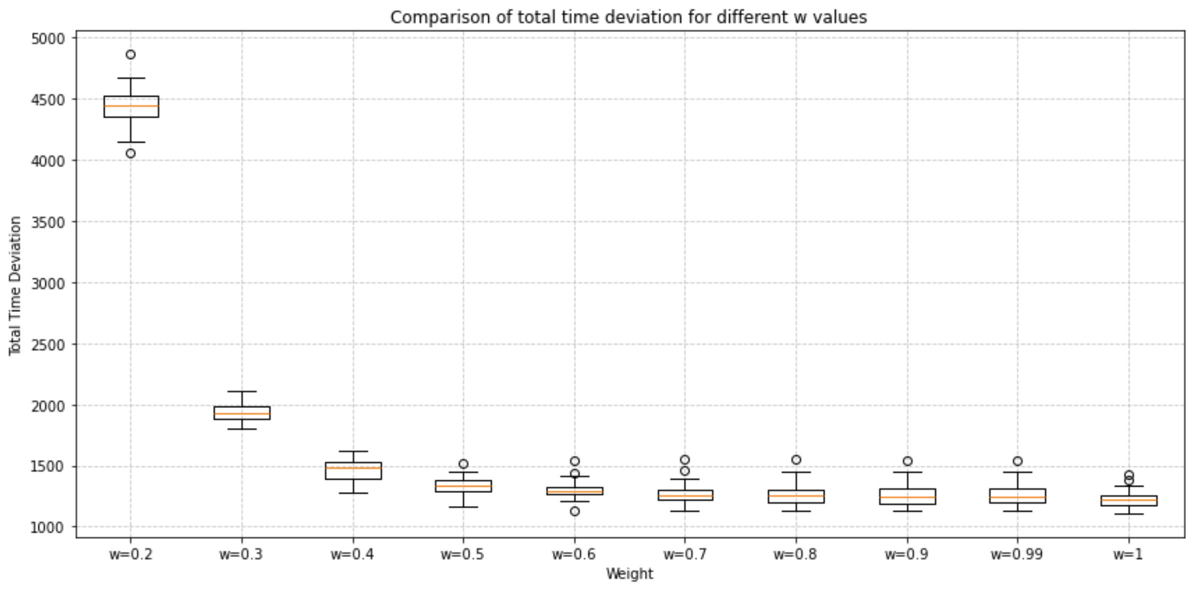


Figure A.23: Capacity = 10 trains per hour, Linear Congestion Factor, Congestion Factor Not Request Dependent

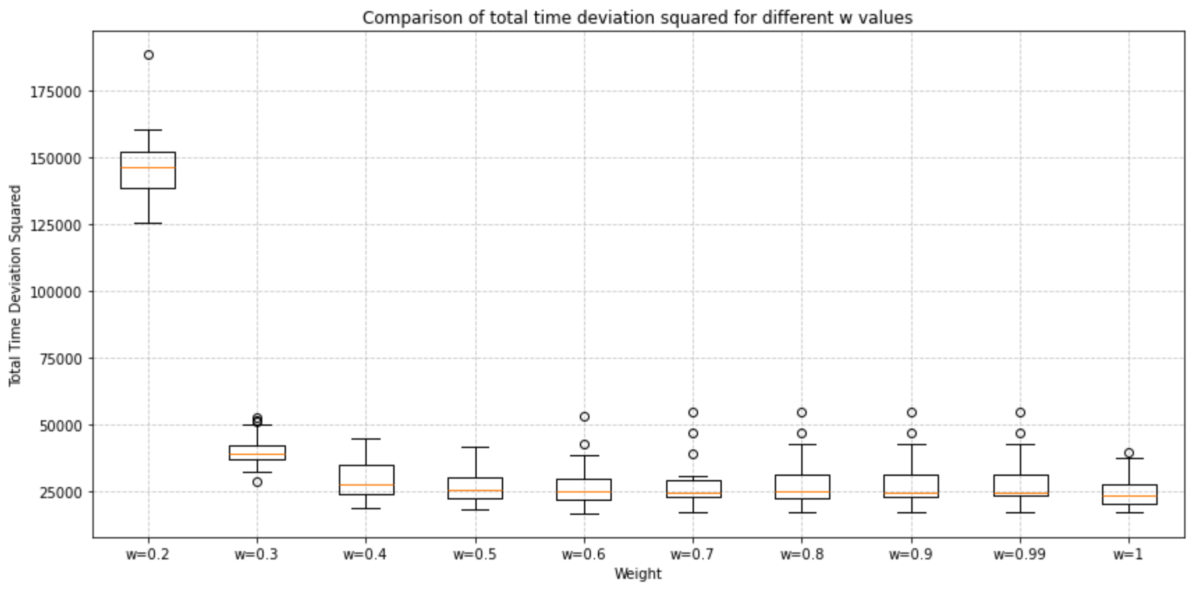


Figure A.24: Capacity = 10 trains per hour, Linear Congestion Factor, Congestion Factor Not Request Dependent, Total Time Deviation Squared

B Tables Repeated Sensitivity Analysis

Below are the results presented from conducting the weighted sensitivity analysis. The results are divided into 4 categories with either the squared congestion factor or normal congestion factor, or scaling the congestion factor down per request or not.

All results in the table are a comparison with the base case scenario where $w = 1$, showing the improvement compared to the base case. That's why there is no row with the weight $w = 1$. The column titles in the tables below mean the following:

- **Weight:** is the weight for which the experiment is conducted [0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 0.99, 1].
- **Average Improvement Time:** When a list of requests is handled, the experiment with the list is conducted for multiple weights. For each weight, the total time deviation after allocating all requests is measured and compared with the total time deviation of the base case scenario ($w = 1$). After running the experiment 50 times, the average improvement is measured compared to the base case scenario.
- **Average Improvement Time Squared:** This is the same as the improvement in total time deviation but considers the squared deviations. It emphasizes larger deviations by penalizing them more heavily, as they are squared before being averaged across all requests and experiments.
- **Percentage Better Result Time:** This measures the percentage of experiments where the total time deviation for a specific weight was lower than that of the base case scenario. It quantifies the frequency of improvement over the base case.
- **Percentage Better Result Time Squared:** Similar to the previous column, this measures the percentage of experiments where the total squared time deviation for a specific weight was lower than that of the base case scenario. It reflects how often a weight achieves better results in terms of departure time deviations squared.

Congestion Factor Sections Squared and Scaling Congestion factor down per Request

Table B.1: Capacity = 5 trains per hour, Squared Congestion Factor, Congestion Factor Request Dependent

Weight (w)	Average Improvement Time	Average Improvement Time Squared	Percentage Better Result Time	Percentage Better Result Time Squared
0.2	-2439.84	-718959.96	8.00%	28.00%
0.3	-1709.58	-936079.38	14.00%	20.00%
0.4	-911.78	-482155.62	24.00%	30.00%
0.5	-856.84	-600581.56	24.00%	30.00%
0.6	-706.58	-609327.42	30.00%	38.00%
0.7	-401.60	-361794.12	34.00%	32.00%
0.8	-209.88	-207314.84	46.00%	44.00%
0.9	35.50	-70297.54	56.00%	54.00%
0.99	100.78	-21761.62	56.00%	54.00%

Table B.2: Capacity = 7.5 trains per hour, Squared Congestion Factor, Congestion Factor Request Dependent

Weight (w)	Average Improvement Time	Average Improvement Time Squared	Percentage Better Result Time	Percentage Better Result Time Squared
0.2	-803.22	-45653.46	0.00%	6.00%
0.3	-376.70	-26603.94	2.00%	18.00%
0.4	-229.80	-21200.56	10.00%	16.00%
0.5	-157.10	-19986.10	18.00%	22.00%
0.6	-107.12	-15771.28	24.00%	24.00%
0.7	-89.86	-15316.94	32.00%	26.00%
0.8	-58.64	-13512.56	38.00%	36.00%
0.9	-25.84	-9349.12	48.00%	36.00%
0.99	-26.70	-9128.58	48.00%	38.00%

Table B.3: Capacity = 10 trains per hour, Squared Congestion Factor, Congestion Factor Request Dependent

Weight (w)	Average Improvement Time	Average Improvement Time Squared	Percentage Better Result Time	Percentage Better Result Time Squared
0.3	-80.44	-3686.84	14.00%	22.00%
0.4	-24.16	-1505.40	40.00%	34.00%
0.5	-17.50	-1583.26	42.00%	36.00%
0.6	-3.58	-1196.70	42.00%	34.00%
0.7	0.26	-1029.98	52.00%	34.00%
0.8	-0.70	-1090.58	54.00%	36.00%
0.9	-1.68	-1144.40	52.00%	34.00%
0.99	-0.80	-1070.32	52.00%	34.00%

Congestion Factor Sections Linear and Scaling Congestion factor down per Request

Table B.4: Capacity = 5 trains per hour, Linear Congestion Factor, Congestion Factor Request Dependent

Weight (w)	Average Improvement Time	Average Improvement Time Squared	Percentage Better Result Time	Percentage Better Result Time Squared
0.2	-4652.02	-1140668.30	2.00%	26.00%
0.3	-1883.06	-863635.66	14.00%	30.00%
0.4	-1190.04	-810594.16	20.00%	30.00%
0.5	-682.32	-592894.16	24.00%	30.00%
0.6	-225.80	-208033.56	40.00%	38.00%
0.7	-297.24	-234580.64	34.00%	30.00%
0.8	-189.32	-232586.52	44.00%	42.00%
0.9	-73.76	-136074.04	48.00%	50.00%
0.99	-77.10	-119702.74	50.00%	52.00%

Table B.5: Capacity = 7.5 trains per hour, Linear Congestion Factor, Congestion Factor Request Dependent

Weight (w)	Average Improvement Time	Average Improvement Time Squared	Percentage Better Result Time	Percentage Better Result Time Squared
0.2	-2305.66	-104314.22	0.00%	0.00%
0.3	-685.20	-33988.04	0.00%	22.00%
0.4	-250.32	-18477.60	8.00%	24.00%
0.5	-125.24	-15378.60	22.00%	32.00%
0.6	-76.00	-12908.08	26.00%	26.00%
0.7	-36.22	-7980.30	38.00%	20.00%
0.8	-22.62	-7476.22	40.00%	22.00%
0.9	2.14	-4893.38	50.00%	38.00%
0.99	4.04	-6221.28	50.00%	40.00%

Table B.6: Capacity = 10 trains per hour, Linear Congestion Factor, Congestion Factor Request Dependent

Weight (w)	Average Improvement Time	Average Improvement Time Squared	Percentage Better Result Time	Percentage Better Result Time Squared
0.2	-1103.34	-40731.02	0.00%	0.00%
0.3	-267.34	-6632.58	0.00%	14.00%
0.4	-85.50	-2632.90	14.00%	42.00%
0.5	-31.44	-1207.84	38.00%	46.00%
0.6	-5.70	-677.58	54.00%	56.00%
0.7	-8.26	-1226.10	56.00%	48.00%
0.8	-7.82	-1324.62	60.00%	48.00%
0.9	-6.26	-1321.46	60.00%	50.00%
0.99	-6.26	-1322.42	60.00%	50.00%

Congestion Factor Sections Squared and Congestion Factor stays constant

Table B.7: Capacity = 5 trains per hour, Squared Congestion Factor, Not Request Dependent

Weight (w)	Average Improvement Time	Average Improvement Time Squared	Percentage Better Result Time	Percentage Better Result Time Squared
0.2	-3947.86	-959281.30	4.00%	36.00%
0.3	-2603.36	-1214142.48	8.00%	22.00%
0.4	-1417.02	-921239.90	24.00%	30.00%
0.5	-1168.74	-887004.02	20.00%	30.00%
0.6	-912.04	-824156.96	26.00%	26.00%
0.7	-568.82	-651972.74	30.00%	28.00%
0.8	-194.76	-302914.08	46.00%	42.00%
0.9	-7.50	-124663.94	44.00%	40.00%
0.99	95.70	-12898.38	54.00%	50.00%

Table B.8: Capacity = 7.5 trains per hour, Squared Congestion Factor, Not Request Dependent

Weight (w)	Average Improvement Time	Average Improvement Time Squared	Percentage Better Result Time	Percentage Better Result Time Squared
0.2	-1293.64	-56766.08	0.00%	10.00%
0.3	-561.96	-31707.64	0.00%	18.00%
0.4	-326.66	-25225.74	6.00%	22.00%
0.5	-201.72	-22833.92	16.00%	20.00%
0.6	-135.08	-20792.80	26.00%	26.00%
0.7	-89.60	-16917.92	28.00%	26.00%
0.8	-47.00	-13500.56	44.00%	32.00%
0.9	-14.90	-9999.22	50.00%	32.00%
0.99	-0.82	-7014.82	50.00%	36.00%

Table B.9: Capacity = 10 trains per hour, Squared Congestion Factor, Not Request Dependent

Weight (w)	Average Improvement Time	Average Improvement Time Squared	Percentage Better Result Time	Percentage Better Result Time Squared
0.2	-384.28	-13531.56	0.00%	4.00%
0.3	-116.44	-3787.64	16.00%	24.00%
0.4	-16.32	-67.12	48.00%	44.00%
0.5	-17.44	-1276.32	36.00%	36.00%
0.6	9.52	-310.24	48.00%	40.00%
0.7	7.44	-638.00	52.00%	40.00%
0.8	8.04	-670.20	56.00%	44.00%
0.9	8.60	-446.20	60.00%	48.00%
0.99	7.28	-511.20	56.00%	48.00%

Congestion Factor Sections Linear and Congestion Factor stays constant

Table B.10: Capacity = 5 trains per hour, Congestion Factor Linear, Not Request Dependent

Weight (w)	Average Improvement Time	Average Improvement Time Squared	Percentage Better Result Time	Percentage Better Result Time Squared
0.2	-10494.24	-888239.36	0.00%	40.00%
0.3	-4329.60	-1236375.20	0.00%	12.00%
0.4	-2292.92	-1209439.24	8.00%	24.00%
0.5	-1755.52	-1331708.24	8.00%	8.00%
0.6	-754.44	-606999.00	24.00%	28.00%
0.7	-452.76	-416548.60	24.00%	20.00%
0.8	-490.08	-414180.80	24.00%	24.00%
0.9	-213.60	-212163.52	32.00%	24.00%
0.99	-112.40	-132329.28	40.00%	32.00%

Table B.11: Capacity = 7.5 trains per hour, Congestion Factor Linear, Not Request Dependent

Weight (w)	Average Improvement Time	Average Improvement Time Squared	Percentage Better Result Time	Percentage Better Result Time Squared
0.2	-6526.92	-281390.04	0.00%	0.00%
0.3	-1882.92	-62378.60	0.00%	4.00%
0.4	-638.12	-20701.08	0.00%	28.00%
0.5	-256.16	-14767.52	12.00%	24.00%
0.6	-164.68	-14734.92	8.00%	28.00%
0.7	-75.48	-7408.12	24.00%	28.00%
0.8	-5.68	-1403.28	40.00%	36.00%
0.9	11.80	-773.64	56.00%	52.00%
0.99	11.28	-1240.72	56.00%	52.00%

Table B.12: Capacity = 10 trains per hour, Congestion Factor Linear, Not Request Dependent

Weight (w)	Average Improvement Time	Average Improvement Time Squared	Percentage Better Result Time	Percentage Better Result Time Squared
0.2	-3203.44	-122108.80	0.00%	0.00%
0.3	-710.28	-16013.64	0.00%	0.00%
0.4	-224.80	-4472.56	4.00%	24.00%
0.5	-104.92	-2528.28	8.00%	24.00%
0.6	-64.04	-2425.24	16.00%	20.00%
0.7	-39.96	-2367.00	16.00%	28.00%
0.8	-38.64	-2785.68	20.00%	24.00%
0.9	-32.68	-2765.16	20.00%	20.00%
0.99	-33.28	-2873.84	24.00%	16.00%

C
Time Space Graphs

In this appendix chapter time-space graphs are plotted to get a better feel of how requests are allocated to itineraries. In Figures C.1, C.2 and C.3 the arc which is most used (Whzan, Brdv, used 86 times) is plotted with all the itineraries using that specific arc.

In Figures C.4 and C.5 all itineraries are plotted in a 3d graph. The idea is that plotting all used itineraries in a 2d graph would be too chaotic. However, visualizing this in a 3d graph does not solve this problem. The main conclusion which can be drawn from plotting the 3d graph is that plotting all freight trains for a single day is too much to visualize in one graph.

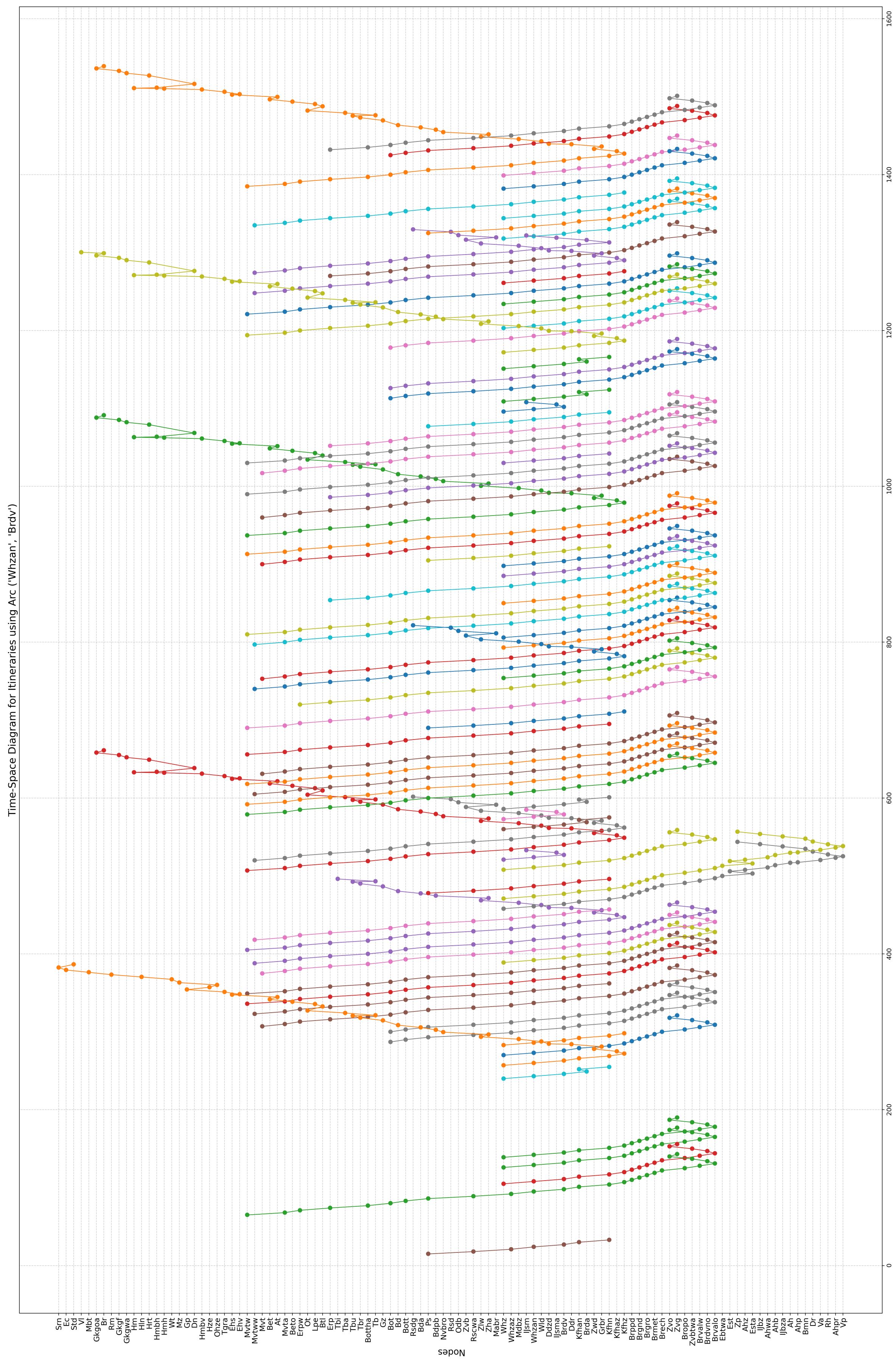


Figure C.1: All allocated itineraries on a time-space graph using the arc (Whzan, Brdv) for w=1

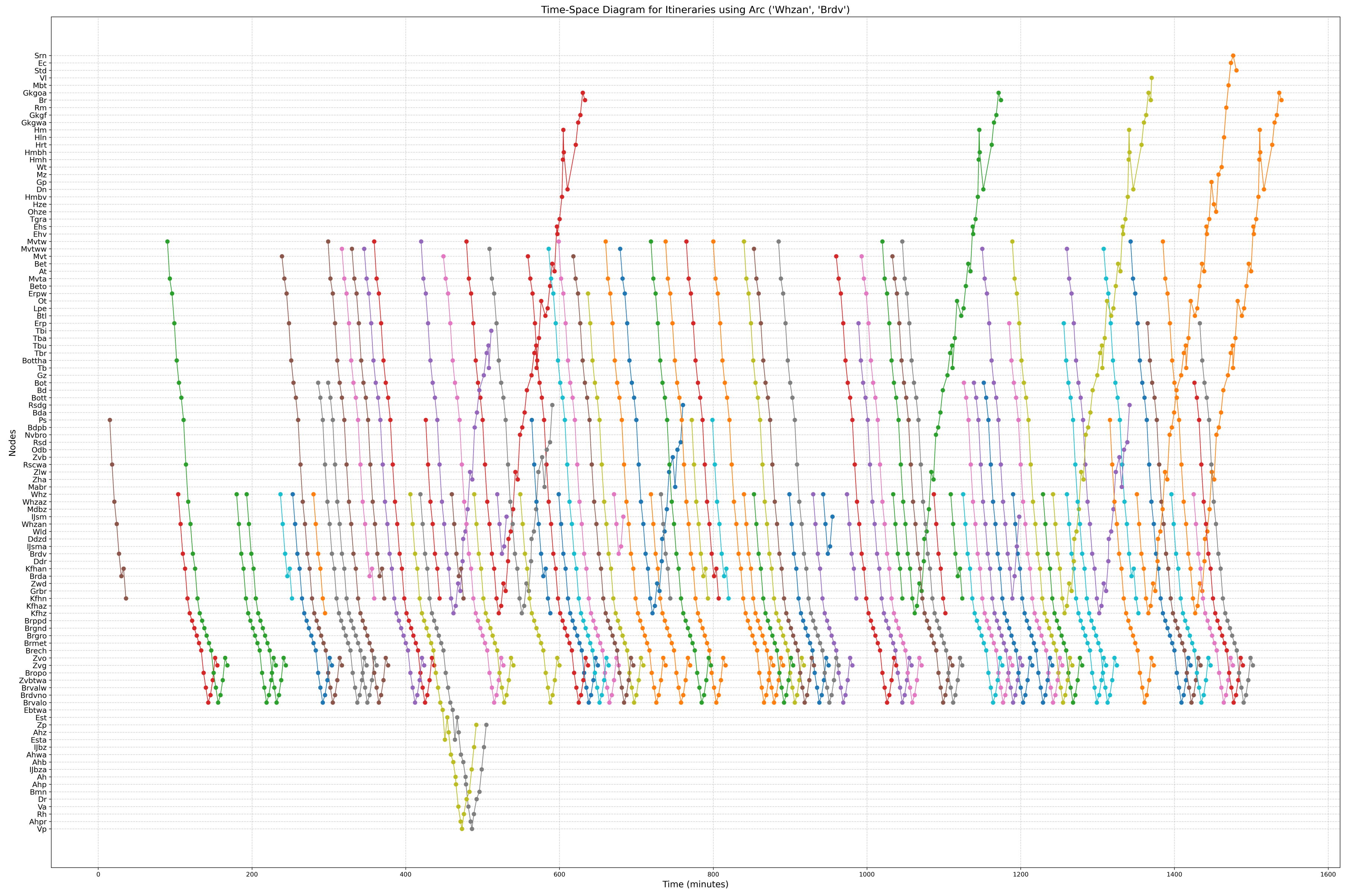


Figure C.2: All allocated itineraries on a time-space graph using the arc (Whzan, Brdv) for w=0.1

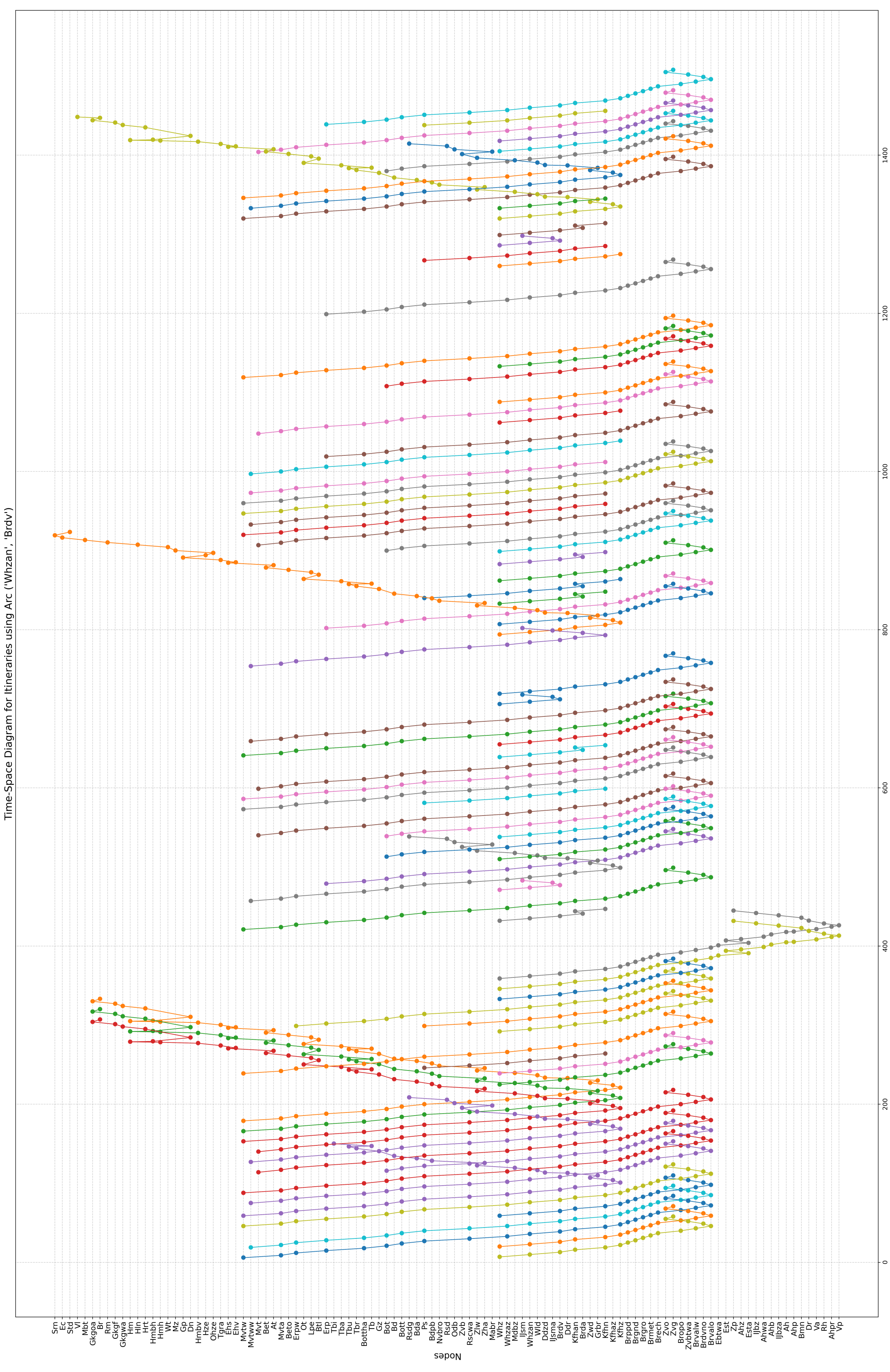


Figure C.3: All allocated itineraries on a time-space graph using the arc (Whzan, Brdv) for w=0

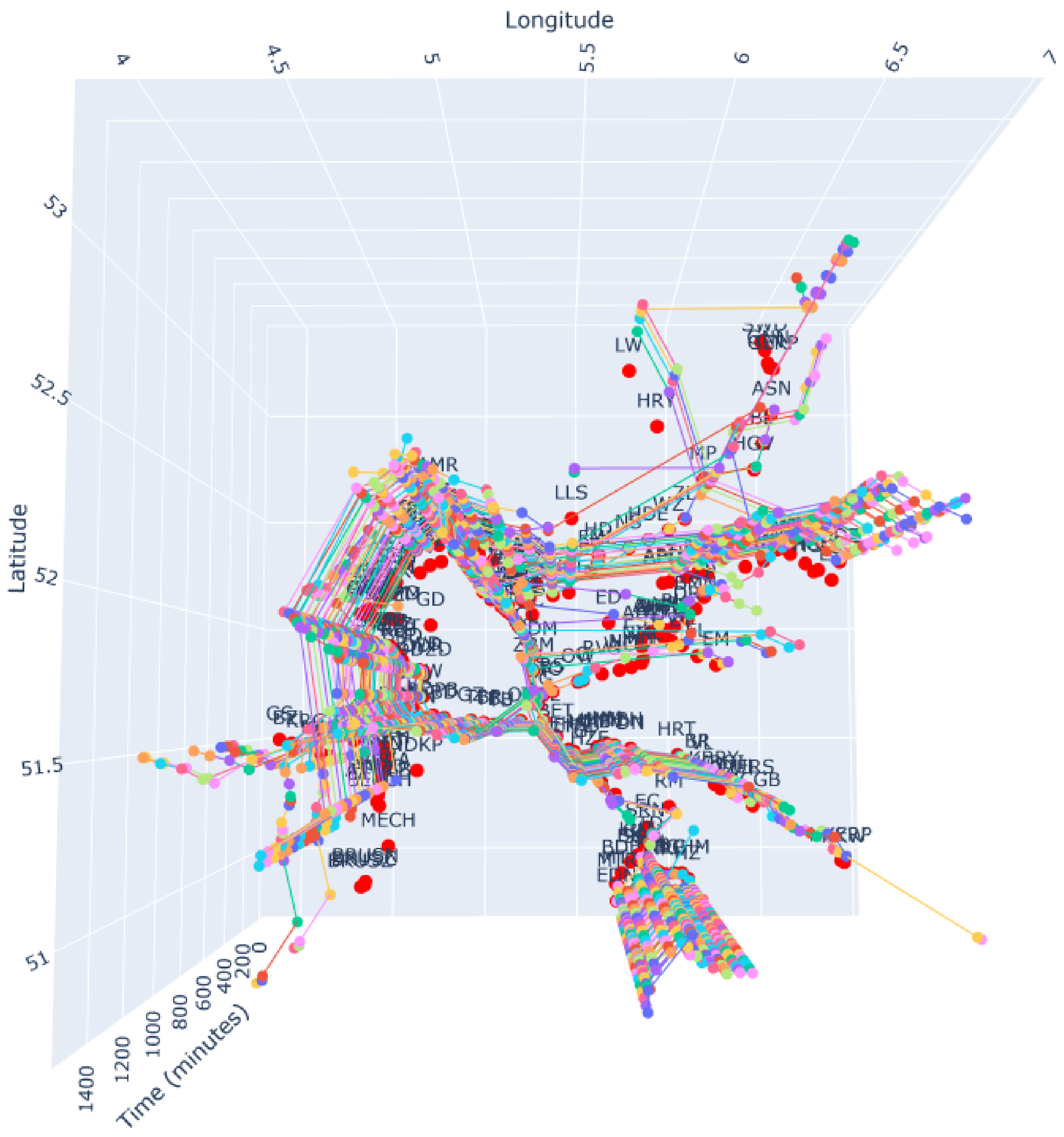


Figure C.4: Allocation of All Requests in a 3d Time Space Graph (top view)

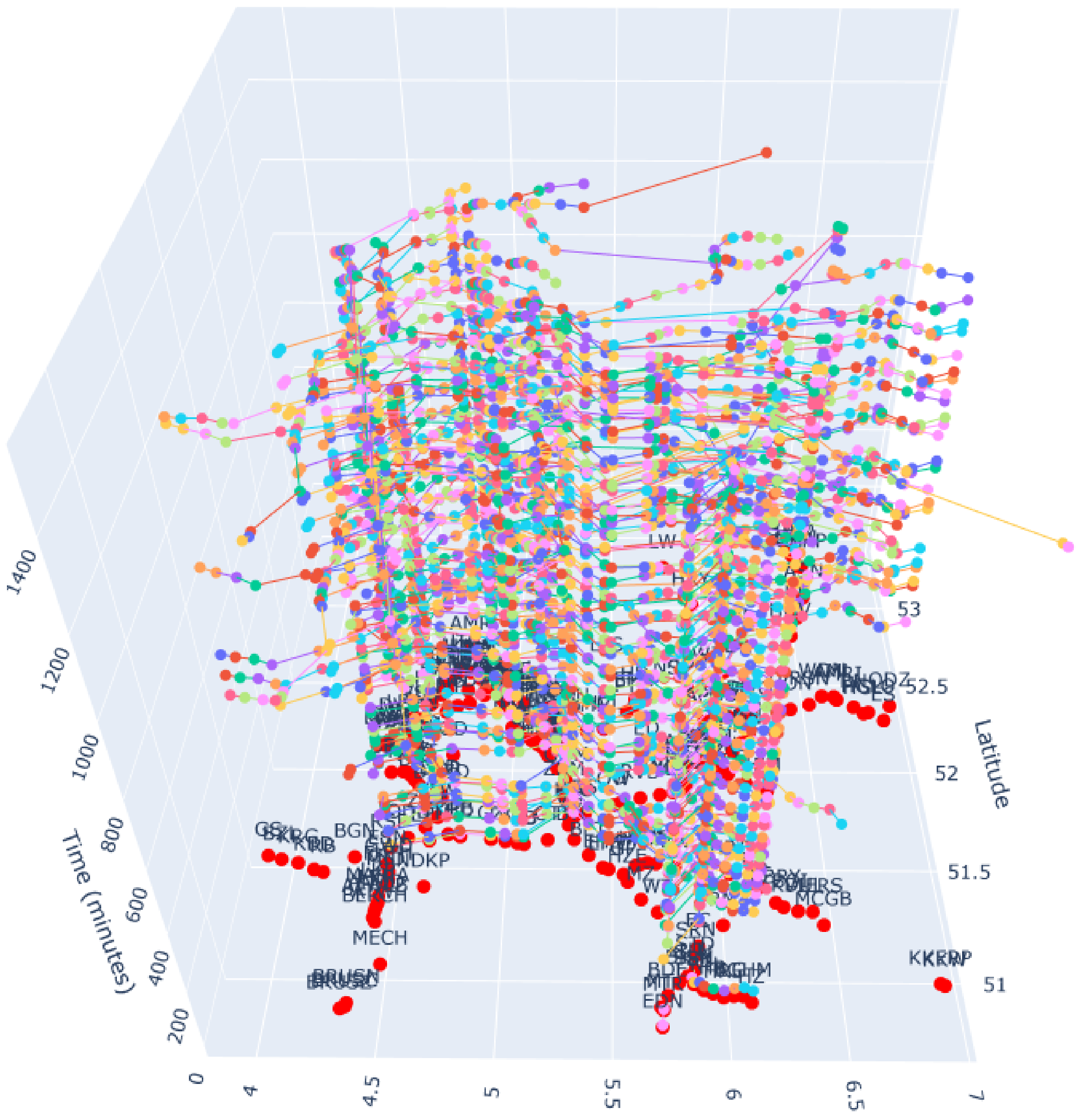


Figure C.5: Allocation of All Requests in a 3d Time Space Graph (side view)

D Allocation of Itineraries in Two Scenario’s

Table D.1: w=1, itinerary allocation, regular request order, first 30 requests

Request	Origin	Dest.	Pref. Time	Path	Dep. Time	Penalty	Time Dev. Sq.	Time Deviation
495740	Whz	Zvg	806	(’Whz’, ’Whzan’, ’Brdv’, ’Kfhan’, ’Kfhn’, ’Kfhz’, ’Brppd’, ’Brgnd’, ’Brgro’, ’Brmet’, ’Brech’, ’Bropo’, ’Brvalw’, ’Brvalo’, ’Brdvno’, ’Zvbtwa’, ’Zvo’, ’Zvg’)	806	2.742	0	0
515241	Mdk	Zlw	1380	(’Mdk’, ’Zlw’)	1380	0.040	0	0
515245	Zlw	Mdk	60	(’Zlw’, ’Mdk’)	60	0.040	0	0
515258	Mdk	Zlw	600	(’Mdk’, ’Zlw’)	600	0.040	0	0
517048	Mvtww	Erp	843	(’Mvtww’, ’Mvta’, ’Erpw’, ’Erp’)	843	0.720	0	0
522430	Mt	Edng	379	(’Mt’, ’Mtr’, ’Edn’, ’Edng’)	379	0.010	0	0
522431	Edng	Mt	327	(’Edng’, ’Mt’)	327	0.010	0	0
522434	Mt	Hang	354	(’Mt’, ’Bha’, ’Luta’, ’Std’, ’Gln’, ’Sbk’, ’Sn’, ’Nh’, ’Hb’, ’Hrla’, ’Hrl’, ’Lg’, ’Eghm’, ’Han’, ’Hang’)	354	0.057	0	0
522436	Edng	Mt	1407	(’Edng’, ’Mt’)	1407	0.010	0	0
522437	Mt	Hang	1434	(’Mt’, ’Bha’, ’Luta’, ’Std’, ’Gln’, ’Sbk’, ’Sn’, ’Nh’, ’Hb’, ’Hrla’, ’Hrl’, ’Lg’, ’Eghm’, ’Han’, ’Hang’)	1434	0.040	0	0
522438	Edng	Mt	1347	(’Edng’, ’Mt’)	1347	0.010	0	0
522439	Mt	Hang	1374	(’Mt’, ’Bha’, ’Luta’, ’Std’, ’Gln’, ’Sbk’, ’Sn’, ’Nh’, ’Hb’, ’Hrla’, ’Hrl’, ’Lg’, ’Eghm’, ’Han’, ’Hang’)	1374	0.040	0	0
522440	Edng	Mt	1287	(’Edng’, ’Mt’)	1287	0.010	0	0

Continued on next page

Table D.1 – continued from previous page

Request	Origin	Dest.	Pref. Time	Path	Dep. Time	Penalty	Dev. Sq.	Deviation
522441	Mt	Hang	1314	('Mt', 'Bha', 'Luta', 'Std', 'Gln', 'Sbk', 'Sn', 'Nh', 'Hb', 'Hrla', 'Hrl', 'Lg', 'Eghm', 'Han', 'Hang')	1314	0.040	0	0
522442	Edng	Mt	1227	('Edng', 'Mt')	1227	0.010	0	0
522443	Mt	Hang	1254	('Mt', 'Bha', 'Luta', 'Std', 'Gln', 'Sbk', 'Sn', 'Nh', 'Hb', 'Hrla', 'Hrl', 'Lg', 'Eghm', 'Han', 'Hang')	1254	0.040	0	0
522444	Edng	Mt	1167	('Edng', 'Mt')	1167	0.010	0	0
522445	Mt	Hang	1194	('Mt', 'Bha', 'Luta', 'Std', 'Gln', 'Sbk', 'Sn', 'Nh', 'Hb', 'Hrla', 'Hrl', 'Lg', 'Eghm', 'Han', 'Hang')	1194	0.040	0	0
522447	Edng	Mt	1107	('Edng', 'Mt')	1107	0.010	0	0
522448	Mt	Hang	1134	('Mt', 'Bha', 'Luta', 'Std', 'Gln', 'Sbk', 'Sn', 'Nh', 'Hb', 'Hrla', 'Hrl', 'Lg', 'Eghm', 'Han', 'Hang')	1134	0.066	0	0
522450	Mt	Hang	1074	('Mt', 'Bha', 'Luta', 'Std', 'Gln', 'Sbk', 'Sn', 'Nh', 'Hb', 'Hrla', 'Hrl', 'Lg', 'Eghm', 'Han', 'Hang')	1074	0.160	0	0
522451	Edng	Mt	1047	('Edng', 'Mt')	1047	0.010	0	0
522452	Edng	Mt	987	('Edng', 'Mt')	987	0.010	0	0
522453	Mt	Hang	1014	('Mt', 'Bha', 'Luta', 'Std', 'Gln', 'Sbk', 'Sn', 'Nh', 'Hb', 'Hrla', 'Hrl', 'Lg', 'Eghm', 'Han', 'Hang')	1014	0.174	0	0
522455	Edng	Mt	927	('Edng', 'Mt')	927	0.010	0	0
522456	Mt	Hang	954	('Mt', 'Bha', 'Luta', 'Std', 'Gln', 'Sbk', 'Sn', 'Nh', 'Hb', 'Hrla', 'Hrl', 'Lg', 'Eghm', 'Han', 'Hang')	954	0.160	0	0
522457	Edng	Mt	867	('Edng', 'Mt')	867	0.010	0	0
522458	Mt	Hang	894	('Mt', 'Bha', 'Luta', 'Std', 'Gln', 'Sbk', 'Sn', 'Nh', 'Hb', 'Hrla', 'Hrl', 'Lg', 'Eghm', 'Han', 'Hang')	894	0.160	0	0
522459	Edng	Mt	807	('Edng', 'Mt')	807	0.010	0	0

Table D.2: w=0.16, itinerary allocation, regular request order, first 30 requests

Request	Origin	Dest.	Pref. Time	Path	Dep. Time	Penalty	Time Dev. Sq.	Time Deviation
495740	Whz	Zvg	806	('Whz', 'Whzan', 'Brdv', 'Kfhan', 'Kfhn', 'Kfhz', 'Brppd', 'Brgnd', 'Brgro', 'Brmet', 'Brech', 'Bropo', 'Brvalw', 'Brvalo', 'Brdvno', 'Zvbtwa', 'Zvo', 'Zvg')	599	0.019	42849	207
515241	Mdk	Zlw	1380	('Mdk', 'Zlw')	1379	0.000	1	1
515245	Zlw	Mdk	60	('Zlw', 'Mdk')	59	0.000	1	1
515258	Mdk	Zlw	600	('Mdk', 'Zlw')	599	0.000	1	1
517048	Mvtww	Erp	843	('Mvtww', 'Mvta', 'Erpw', 'Erp')	900	0.107	3249	57
522430	Mt	Edng	379	('Mt', 'Edng')	379	0.010	0	0
522431	Edng	Mt	327	('Edng', 'Edn', 'Mtr', 'Mt')	327	0.010	0	0
522434	Mt	Hang	354	('Mt', 'Bha', 'Luta', 'Std', 'Gln', 'Sbk', 'Sn', 'Nh', 'Hb', 'Hrla', 'Hrl', 'Lg', 'Eghm', 'Han', 'Hang')	353	0.054	1	1
522436	Edng	Mt	1407	('Edng', 'Edn', 'Mtr', 'Mt')	1407	0.010	0	0
522437	Mt	Hang	1434	('Mt', 'Bha', 'Luta', 'Std', 'Gln', 'Sbk', 'Sn', 'Nh', 'Hb', 'Hrla', 'Hrl', 'Lg', 'Eghm', 'Han', 'Hang')	1434	0.040	0	0
522438	Edng	Mt	1347	('Edng', 'Edn', 'Mtr', 'Mt')	1347	0.010	0	0
522439	Mt	Hang	1374	('Mt', 'Bha', 'Luta', 'Std', 'Gln', 'Sbk', 'Sn', 'Nh', 'Hb', 'Hrla', 'Hrl', 'Lg', 'Eghm', 'Han', 'Hang')	1374	0.040	0	0
522440	Edng	Mt	1287	('Edng', 'Edn', 'Mtr', 'Mt')	1287	0.010	0	0
522441	Mt	Hang	1314	('Mt', 'Bha', 'Luta', 'Std', 'Gln', 'Sbk', 'Sn', 'Nh', 'Hb', 'Hrla', 'Hrl', 'Lg', 'Eghm', 'Han', 'Hang')	1314	0.040	0	0
522442	Edng	Mt	1227	('Edng', 'Edn', 'Mtr', 'Mt')	1227	0.010	0	0
522443	Mt	Hang	1254	('Mt', 'Bha', 'Luta', 'Std', 'Gln', 'Sbk', 'Sn', 'Nh', 'Hb', 'Hrla', 'Hrl', 'Lg', 'Eghm', 'Han', 'Hang')	1254	0.040	0	0
522444	Edng	Mt	1167	('Edng', 'Edn', 'Mtr', 'Mt')	1167	0.010	0	0
522445	Mt	Hang	1194	('Mt', 'Bha', 'Luta', 'Std', 'Gln', 'Sbk', 'Sn', 'Nh', 'Hb', 'Hrla', 'Hrl', 'Lg', 'Eghm', 'Han', 'Hang')	1194	0.040	0	0
522447	Edng	Mt	1107	('Edng', 'Edn', 'Mtr', 'Mt')	1107	0.010	0	0
522448	Mt	Hang	1134	('Mt', 'Bha', 'Luta', 'Std', 'Gln', 'Sbk', 'Sn', 'Nh', 'Hb', 'Hrla', 'Hrl', 'Lg', 'Eghm', 'Han', 'Hang')	1140	0.043	36	6

Continued on next page

Table D.2 – continued from previous page

Request	Origin	Dest.	Pref. Time	Path	Dep. Time	Penalty	Dev. Sq.	Deviation
522450	Mt	Hang	1074	(’Mt’, ’Bha’, ’Luta’, ’Std’, ’Gln’, ’Sbk’, ’Sn’, ’Nh’, ’Hb’, ’Hrla’, ’Hrl’, ’Lg’, ’Eghm’, ’Han’, ’Hang’)	1153	0.004	6241	79
522451	Edng	Mt	1047	(’Edng’, ’Edn’, ’Mtr’, ’Mt’)	1047	0.010	0	0
522452	Edng	Mt	987	(’Edng’, ’Edn’, ’Mtr’, ’Mt’)	987	0.010	0	0
522453	Mt	Hang	1014	(’Mt’, ’Bha’, ’Luta’, ’Std’, ’Gln’, ’Sbk’, ’Sn’, ’Nh’, ’Hb’, ’Hrla’, ’Hrl’, ’Lg’, ’Eghm’, ’Han’, ’Hang’)	1080	0.040	4356	66
522455	Edng	Mt	927	(’Edng’, ’Edn’, ’Mtr’, ’Mt’)	927	0.010	0	0
522456	Mt	Hang	954	(’Mt’, ’Bha’, ’Luta’, ’Std’, ’Gln’, ’Sbk’, ’Sn’, ’Nh’, ’Hb’, ’Hrla’, ’Hrl’, ’Lg’, ’Eghm’, ’Han’, ’Hang’)	960	0.143	36	6
522457	Edng	Mt	867	(’Edng’, ’Edn’, ’Mtr’, ’Mt’)	867	0.010	0	0
522458	Mt	Hang	894	(’Mt’, ’Bha’, ’Luta’, ’Std’, ’Gln’, ’Sbk’, ’Sn’, ’Nh’, ’Hb’, ’Hrla’, ’Hrl’, ’Lg’, ’Eghm’, ’Han’, ’Hang’)	900	0.143	36	6
522459	Edng	Mt	807	(’Edng’, ’Edn’, ’Mtr’, ’Mt’)	807	0.010	0	0

E

Load Distribution on the Network

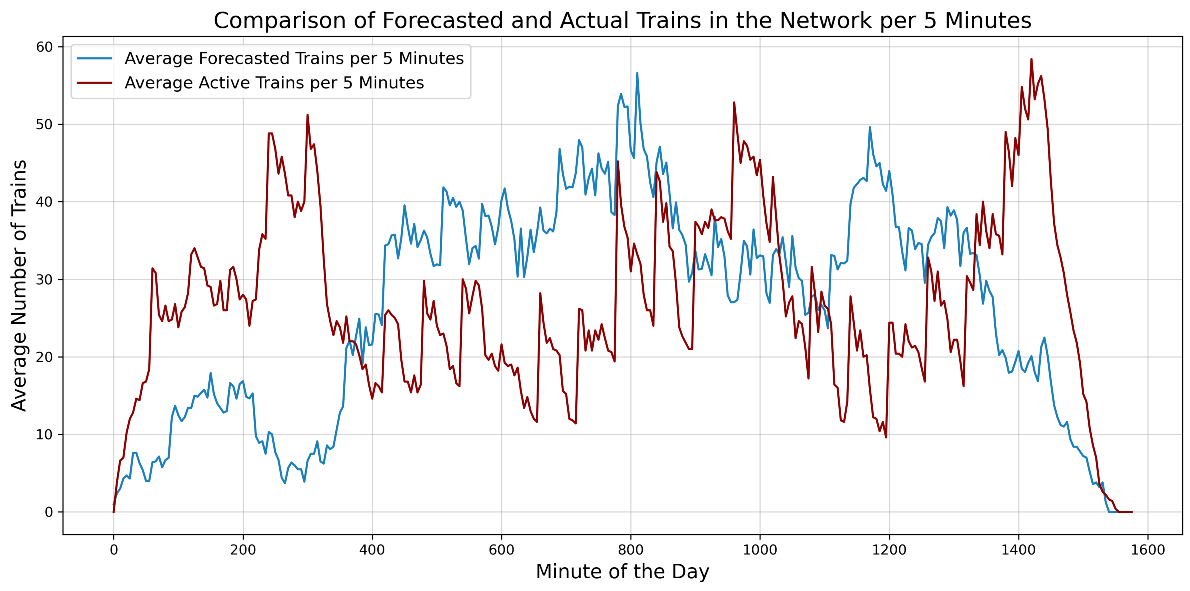


Figure E.1: Average forecasted trains per 5 minutes for $w = 0$, capacity=5 trains per hour

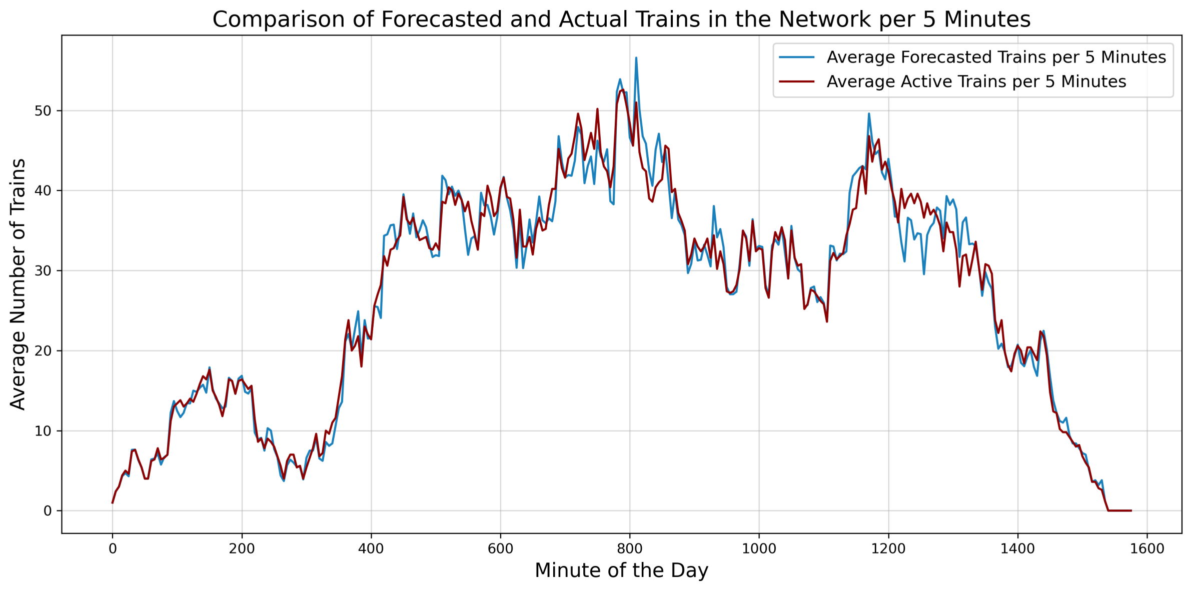


Figure E.2: Average forecasted trains per 5 minutes for $w = 1$, capacity=10 trains per hour

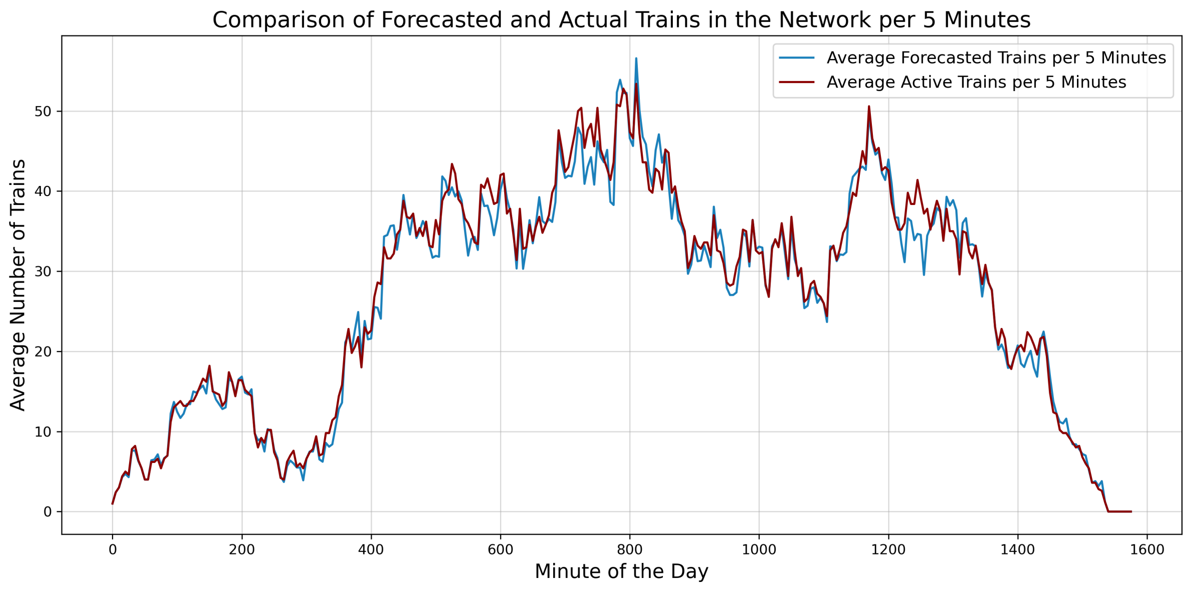


Figure E.3: Average forecasted trains per 5 minutes for $w = 0.6 - 0.99$, capacity=10 trains per hour

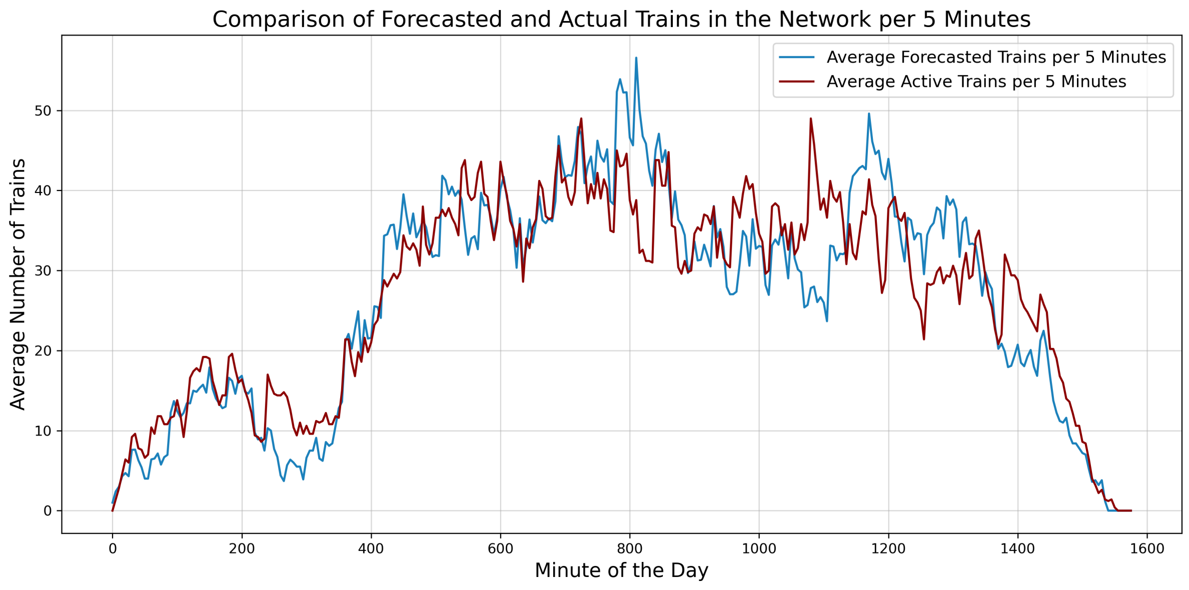


Figure E.4: Average forecasted trains per 5 minutes for $w = 0.15$, capacity=5 trains per hour, reversed order

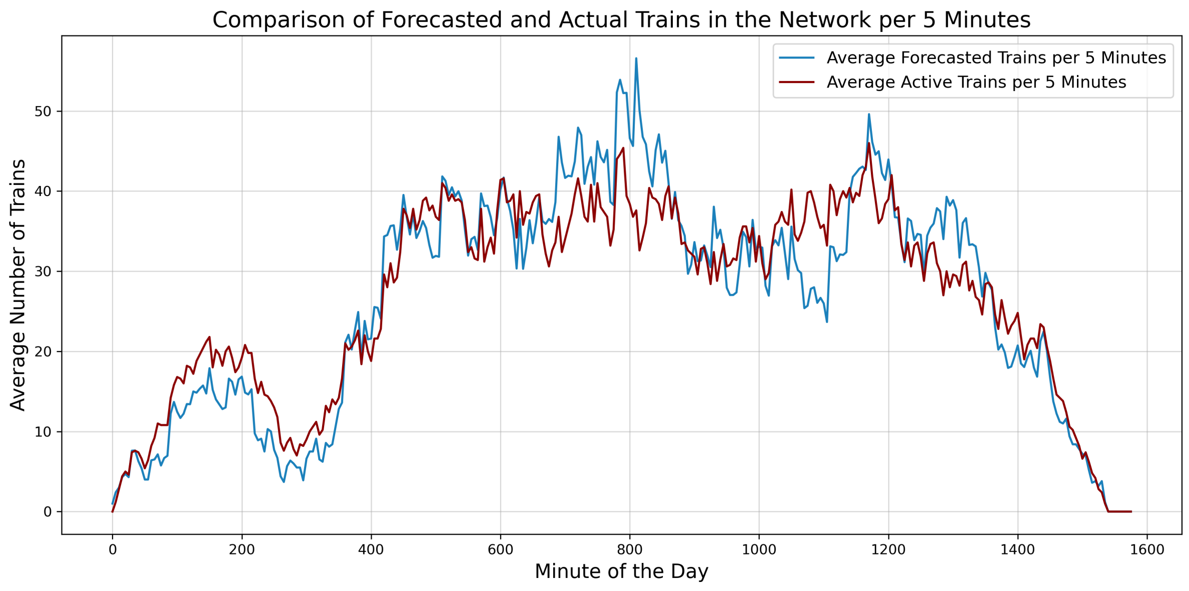


Figure E.5: Average forecasted trains per 5 minutes for $w = 1$, capacity=5 trains per hour, reversed order