Università degli Studi di Milano - Bicocca
Dipartimento di Informatica, Sistemistica e Comunicazione
Corso di Laurea in Informatica

Delay Prediction in Supply Chains: A Hybrid Graph-Based and Machine Learning Approach

Relatore: Prof. Michele Ciavotta

Correlatore: Ing. Bruno Puzzolante

Tesi di Laurea di:
Mattia Volpato
Matricola 866316

Anno Accademico 2024-2025

Acknowledgments

While all the content of this work was developed by the author, a meaningful contribution by Eng. Andrea Landella proved significant in the formulation of specific ideas and the design of particular solutions presented in this thesis.

Limited use of generative AI tools was employed during the preparation of this work to support the refinement of clarity, grammar and structure, as well as to retrieve information and inspire the development of ideas for presentation and organization.

All conceptual content, analysis, and conclusions presented in this thesis remain fully the responsibility of the author.

Abstract

Global supply chains face frequent disruptions from both external events, such as natural disasters, and internal operational issues, including supplier and shipment delays. This work, carried out within the scope of the Horizon Europe project *Industrial Manufacturing As a sErvice STRategies and models for flexible, resilient, and reconfigurable value networks through Trusted and Transparent Operations* (**M4ESTRO**), contributes to strengthening manufacturing resilience through real-time disruption monitoring, with a specific focus on internal delivery processes.

We present a mathematical framework for representing the delivery stages of supply chains, supported by lightweight predictive models built on a set of real-time indicators. These indicators quantify operational performance during dispatch and shipment processes, enabling the generation of reliable forecasts and associated uncertainty estimates. The framework leverages probabilistic methods and Markov chain based approaches to estimate delivery times, detect deviations from planned schedules and provide interpretable measures for operational decision-making.

A prototype implementation validates the approach by integrating heterogeneous data sources, including traffic conditions, weather information and supplier production calendars. Results show that monitoring delivery stages through indicators provides actionable insights and enables early disruption detection, offering a transparent and interpretable basis for strengthening resilience in internal operations.

Contents

List of Figures

List of Tables

List of Algorithms

Acronyms

DLT Delivery Time. 40

DRT Dynamic Route Time. 8, 46, 48, 53, 65, 66, 84, 87, 88, 93, 94

DT Dispatch Time. 38, 39, 43

E-PT Expected Path Time. 51, 110

E-TFST Expected Total Forecasted Shipment Time. 51, 52, 110

EDT Estimated Dispatch Time. 42, 43, 52

EODT Estimated Overall Delivery Time. 52

LCDIs Logistics Chain Disruption Indicators. 12, 13, 15, 92, 94

ORI Overall Residence Index. 43–45, 48, 58, 60

OTI Overall Transit Index. 45, 48, 60, 67

PT Path Time. 47, 49–51, 73–77, 81, 83, 84, 86, 91

RT Route Time. 46, 49, 65, 67–70

SCGraph Supply Chain Graph. 34–36, 79, 80, 100–102, 105, 110

ST Shipment Time. 39

TFST Total Forecasted Shipment Time. 49, 73, 74, 76, 82, 84

TMI Traffic Meta Index. 6, 14, 45, 60–62, 67

TT Transit Time. 47, 49–51, 73–77, 81, 83, 84, 86, 104

VT Vertex Time. 44, 46, 48, 49

WMI Weather Meta Index. 6, 14, 45, 62, 64, 65, 67, 94

Chapter 1

Introduction

1.1 Background and Context

The manufacturing industry is undergoing a profound transformation, driven by the need to respond to increasingly frequent and severe disruptions. Events such as the COVID-19 pandemic, geopolitical tensions, and climate-related disasters have exposed the vulnerabilities of global supply chains. In this volatile, uncertain, complex, and ambiguous (VUCA) environment, resilience has become a strategic imperative.

To address these challenges, the **M4ESTRO** project – *Industrial Manufacturing As a sErvice STRategies and models for flexible, resilient, and reconfigurable value networks through Trusted and Transparent Operations* – was launched under the Horizon Europe framework. Coordinated by **Cefriel** [43], a digital innovation center based in Milan, Italy, the M4ESTRO project brings together a consortium of 18 partners–including industrial pilots, technology providers, and research institutions–from 8 different countries [45]. The initiative, launched on 1 December 2023, spans a duration of 42 months and aims to develop a resilient, transparent, and adaptive manufacturing ecosystem [46].

M4ESTRO aims to develop a comprehensive platform for resilient manufacturing, built on the *Manufacturing-as-a-Service* (MaaS) paradigm. The platform is designed to enable active and predictive resilience, offering timely preparedness and adaptive responses to disruptive events. Its architecture is structured around four strategic pillars:

- Resilient, transparent, and flexible manufacturing processes;
- Resilient equipment, AI, and trusted data for adaptive manufacturing;
- Resilient simulations integrated with the Industrial Metaverse;
- Human-centered manufacturing resilience and sustainability.

These pillars support the development of the M4ESTRO Supply Resilience App, which integrates data from diverse sources to monitor supply chain health and suggest corrective actions in response to disruptions.

1.2 Motivation

Despite the growing awareness of supply chain vulnerabilities, most existing optimization approaches are designed for stable or moderately variable conditions. They typically rely on historical data and probabilistic models to manage fluctuations in supply and demand, pricing, and inventory levels. However, these methods often lack the agility and foresight required to respond to sudden, large-scale disruptions.

Furthermore, many current solutions focus on post-event analysis through simulations and scenario planning. While useful for strategic decision-making, they do not provide the real-time situational awareness necessary for operational resilience. In contrast, M4ESTRO introduces a proactive framework for sensing disruptions and enabling adaptive responses.

A key component of this framework is the development of **disruption indicators**, consisting in measurements which monitor and quantify the operational dynamics of the supply chain. These indicators are motivated by the need to:

1. Detect anomalies in real time across logistics and production systems;
2. Quantify the impact of disruptions using data-driven metrics;
3. Enable timely and actionable responses through integration with decision-support tools.

From a research perspective, this task involves challenges such as selecting meaningful metrics, integrating heterogeneous data sources, and designing robust detection mechanisms. Addressing these challenges contributes to both the success of the M4ESTRO platform and the advancement of supply chain resilience methodologies.

1.3 Purpose and Scope

The M4ESTRO project is structured around a set of interconnected tasks, each addressing different aspects of manufacturing resilience. Although this thesis focuses on a specific task, its scope is framed by the broader project activities, which provide both input and context. The most relevant tasks are:

- **T1.1 – External Disruption Indicators**: led by Atlantis Engineering AE [41], this task focuses on identifying and predicting external disruptions using heterogeneous data sources such as financial markets, social and news media, and cargo networks. Techniques such as anomaly detection, rule-based analysis, and machine learning (e.g., natural language processing for sentiment analysis) are employed to estimate disruption likelihood.
- **T1.2 – Internal Disruption Indicators and Reconfiguration Models**: led by Cefriel [43], this task is the central focus of this thesis. It involves the definition and implementation of **Logistics Chain Disruption Indicators (LCDIs)** based on operational data from supply chain stakeholders, including warehouses, production systems and IoT devices, with the aim of enabling the early detection of internal disruptions. In

addition, it addresses the development of reconfiguration models that propose mitigation strategies to reduce disruption impact and response times.

- **T1.4 – AI-Optimised Manufacturing Networks and Resilience Predictor**: led by CORE Innovation Centre [44], this task builds upon the outputs of T1.1, T1.2 and other tasks to develop an AI-based optimization engine. The engine continuously learns from historical data to rank supply chain alternatives according to resilience and performance criteria, providing traceable and adaptive recommendations to support decision-making.

Within this framework, this thesis contributes specifically to **Task 1.2**, which pursues three overarching objectives:

1. the design of **LCDIs** for monitoring supply chain health;
2. the development of a reconfiguration model to support adaptive responses to disruptions.
3. the delivery of a preliminary prototype that operationalizes these models within the M4ESTRO platform.

The work was conducted during a six-month research internship at Cefriel, where I collaborated on both components of Task 1.2 with the team of Smart Industry Solutions. However, the scope of this work is limited to the first objective, namely the design, formalization, and implementation of a selected subset of LCDIs.

Since the project was still in an early phase of its 48-month timeline, integration with pilot partners and the availability of operational data were limited. This scarcity of high-quality data strongly influenced the methodological choices made in this work. In particular, the approach aimed to balance, on the one hand, the adoption of complex models that would require substantial data for calibration, and, on the other hand, the need to deliver a functional prototype within the six-month deadline. As a result, the contribution emphasizes a rigorous formalization of the supply chain framework, while the implementation of its components in some cases relies on pragmatic solutions that will require refinement as richer datasets become available through deeper integration with pilot partners.

1.4 Contributions

This thesis focuses on the construction of a mathematical framework for representing supply chain processes and on the definition of quantitative indicators to measure delivery times and delays. These indicators are designed to enable the early detection of logistic disruptions and to provide actionable input for higher-level resilience mechanisms within the M4ESTRO platform.

When I joined the project, Task 1.2 had already entered a preliminary phase, which outlined the following requirements for the LCDIs definitions:

1. An indicator to quantify the time required for a supplier to complete the dispatching process of an order.

2. An indicator to quantify the time required for a carrier to complete the shipment process of an order.

3. The integration of real-time traffic and weather information into shipment time estimations, expressed through two indicators:

   - Traffic Meta Index (TMI);
   - Weather Meta Index (WMI).

4. The integration of suppliers' holiday calendars into dispatch time estimation, as highlighted by pilot partners as a major source of disruptions.

5. The definition of two reliability indicators, measuring supplier performance (quality issues in delivered materials) and carrier performance (lost shipments).

The contribution of this thesis addresses requirements (1)–(4). Requirement (5), although part of the broader supply chain formalization, falls outside the scope of this work since it is not directly related to delivery time estimation.

Finally, the framework and prototype developed in this thesis are designed for a single-manufacturer supply chain. Extending the approach to multi-manufacturer environments is left for future work.

1.5 Terminology

Throughout this thesis, the following terminology will be adopted for clarity and consistency:

- **Manufacturer**: the end-user of the M4ESTRO platform, responsible for producing goods and interested in improving the resilience of its supply chain.
- **Supplier**: an external company that provides raw materials, components, or semi-finished products to the manufacturer.
- **Site**: the physical facility where a supplier operates. A supplier may manage multiple sites located in different regions worldwide.
- **Carrier**: a logistics service provider responsible for transporting materials from suppliers to the manufacturer.
- **Dispatch**: the process of preparing and releasing an order at the supplier's site.
- **Shipment**: the physical transportation of goods from the supplier's location to the manufacturer's site.
- **Delivery**: the complete end-to-end process combining dispatch and shipment, from order preparation to arrival at the manufacturer.
- **Disruption**: any unexpected event that interferes with the normal execution or organization of supply chain operations, potentially leading to delays, inefficiencies, or failures.

1.6 Assumptions

The design of the framework relies on a set of assumptions that constrain the behavior of suppliers, carriers, and other supply chain actors. These assumptions are informed either by insights from pilot partners or by evidence found in the literature. For clarity, each assumption will be presented and discussed within the section where it directly influences the proposed solution.

1.7 Structure of the Thesis

This thesis is organized into eight chapters:

- **Chapter 1** introduces the work, providing background, motivation, purpose and scope, and summarizing the main contributions.
- **Chapter 2** reviews the literature on supply chain modeling approaches. It integrates both the foundational research conducted in the early stages of the project and the additional sources consulted after joining the project, establishing the theoretical basis for this work.
- **Chapter 3** describes the data provided by the pilot partners, as well as the external data sources utilized.
- **Chapter 4** formalizes the supply chain framework and defines LCDIs, organized into two categories: *historical* and *realtime* indicators.
- **Chapter 5** presents a preliminary implementation of both historical and realtime indicators, developed under the constraints of limited data availability.
- **Chapter 6** provides an overview of the developed prototype, detailing the data model design and the overall system architecture.
- **Chapter 7** reports the results obtained by applying the prototype to real operational data from the pilot partners.
- **Chapter 8** concludes the thesis, summarizing the contributions and outlining directions for future research and development.

Chapter 2

Literature Review

In this chapter, we review the state of the art in supply chain disruption modeling and risk mitigation. We examine the integration of predictive analytics, artificial intelligence (AI), digital twins, and real-time data streams, alongside probabilistic and stochastic models for uncertainty management. In particular, we focus on works that address delay estimation, shipment and dispatch modeling, and the use of uncertainty intervals and Markov chains in supply chain analysis. This review highlights both the progress achieved in the literature and the gaps that motivate the research carried out in this thesis.

2.1 State of the Art

The literature on supply chain risk management reflects a progressive shift from static planning approaches to dynamic, data-driven frameworks. Several trends emerge:

Predictive Analytics

Predictive analytics has become one of the most widely adopted tools for disruption detection and delay estimation. Time-series forecasting, regression models, and anomaly detection are used to anticipate risks based on historical and streaming data. Aljohani [26] provides an extensive survey of such techniques, highlighting applications of supervised and unsupervised learning to real-time risk mitigation. Similarly, Bodendorf et al. [27] combine causal inference with deep learning to identify potential disruption drivers. Recent advances also consider deep architectures such as Graph Neural Networks (Wasiac et al. [29]) and inductive graph transformers (Zhou et al. [30]), which exploit network structure to enhance delivery time estimation and supply chain optimization.

Digital Twins and Simulation

Digital twins are increasingly used to build virtual replicas of supply chains. These enable simulation of disruption scenarios and evaluation of mitigation strategies through computer-based simulations. For instance, virtual models of logistics networks can simulate route blockages or capacity shortages, allowing

decision makers to assess alternative configurations before committing to real-world changes. This mirrors the approach of simulation-based studies such as Seco and Vieira [18], who apply multi-agent models to explore supply chain dynamics under different conditions. Such digital twin methods support proactive disruption management but are often data and computation intensive.

Real-Time Data Integration

Modern supply chains increasingly exploit heterogeneous data sources to improve disruption detection. IoT devices, blockchain-enabled traceability systems and even social media provide real-time visibility into operations. Sallam et al. [28] describe how IoT integration improves transparency and enables continuous monitoring of shipments. This trend underscores a growing reliance on real-time integration for agile decision making.

Uncertainty and Probabilistic Modeling

Uncertainty is inherent in logistics processes, particularly in dispatch and shipment times. Stochastic and probabilistic models are commonly applied to represent variability and to support robust decision making. Carbonneau et al. [13] demonstrate the use of machine learning for demand forecasting under uncertain conditions. Maggioni et al. [20] propose a scenario-based framework for supply planning that compares stochastic programming against robust optimization using simple uncertainty sets, such as box or ellipsoidal sets, built from limited or partial data. This low-data and interpretable approach allows decision makers to hedge against worst-case outcomes without full probabilistic knowledge. This highlights the importance of uncertainty-aware models, particularly lightweight ones, in estimating delays and delivery performance when operational data are sparse.

Markov Chains in Supply Chains

Markov chains constitute a foundational stochastic modeling framework for capturing uncertainty in logistics and supply chain operations. Their ability to represent state transitions probabilistically makes them particularly suitable for modeling shipment progress, disruption propagation, or recovery behaviors. Borucka et al. [24] employ Hidden Markov Models (HMMs) to analyze supply sequences and forecast delivery reliability. By inferring latent disruption states, their approach supports early detection of risk patterns embedded in observed shipment data. Liu et al. [25] propose a robust dynamic Bayesian network that incorporates Markovian transitions to model ripple effects of disruptions. This hybrid framework illustrates how Markov processes can underpin more complex probabilistic reasoning in supply chain resilience modeling. Larin [32] applies a discrete-time Markov chain to quantify supply chain resilience. Transition probabilities between disrupted and recovered states lead to clear, interpretable resilience indicators. Putthakosa and Hirotani [31] present a discrete-time Markov chain model of disruption dynamics in multi-state supply chains. By estimating steady-state probabilities, they enable the evaluation of expected disruption costs and the impact of disruption transitions on supply chain stability.

Together, these studies demonstrate how Markovian methods can effectively model uncertainty, shipment trajectories, and resilience—even under data scarcity and scheduling complexity—making them a valuable tool for contexts similar to our scenario, where operational data is limited.

2.2 Limitations of Existing Approaches

While the literature offers a wide variety of modeling techniques, several limitations emerge when considering their applicability to internal disruptions such as dispatching delays or carrier underperformance:

- **Data Intensity:** Predictive analytics, digital twins, and advanced stochastic models often assume access to extensive and high-quality datasets. In practice, operational data from suppliers and carriers may be scarce or fragmented, limiting the applicability of such approaches.
- **Emphasis on External Risks:** Existing studies predominantly examine external shocks, such as geopolitical crises or natural disasters, while comparatively fewer address operational delays arising from internal supply chain processes. Filling this gap is the main focus of our work.
- **Computational Demands:** Simulation-heavy methods (e.g., digital twins) require substantial resources, which may be impractical for real-time logistics monitoring.
- **Limited KPI Integration:** Although probabilistic models capture uncertainty, they often remain abstract and are not easily translated into interpretable key performance indicators (KPIs).

These constraints make many state-of-the-art methods difficult to adopt in real-world scenarios characterized by low data availability and the need for interpretable, real-time indicators. This limitation was particularly relevant in the context of the M4ESTRO project: at the time of this research, integration with pilot partners was still in an early phase, and only limited operational data were accessible. Consequently, a lightweight modeling strategy was not only methodologically appropriate, but also necessary to ensure feasibility and deliver a functioning prototype within the project's constraints.

2.3 Alternative Modeling Strategies

Given the limitations, lightweight alternatives have emerged. Indicator-based approaches, which rely on a limited set of measurable variables, offer a pragmatic solution for real-time logistics monitoring.

KPI-Driven Approaches

Studies such as Carbonneau et al. [13] and Seco and Vieira [18] demonstrate that even simplified models can provide valuable insights when represented through performance metrics. By focusing on dispatch times, shipment durations, and

their associated variability, researchers can build interpretable indicators of internal disruptions without requiring complete visibility into the entire supply chain.

Uncertainty Intervals as Indicators

Several works employ uncertainty intervals or bounded uncertainty sets to quantify variability in lead times and supply performance. Maggioni et al. [20] show how simple box or ellipsoidal sets can be constructed from limited data to support robust optimization in supply planning. Applying similar concepts to dispatch and shipment delays offers a straightforward yet effective way to monitor disruptions arising from internal operations, even when probabilistic distributions are difficult to estimate.

Markov Chains for Shipments Processes

Markov chains and Hidden Markov Models (HMMs) offer a lightweight approach to modeling supply chain processes. By representing shipment trajectories as sequences of stochastic state transitions, these models can estimate the likelihood of delays or failures even when data are sparse. Such an approach allows calibration on limited historical shipment records while still producing meaningful forecasts of delivery times and potential disruptions along the logistics path.

2.4 Research Gaps and Motivation

From this review, two research gaps emerge:

1. **Limited Attention to Internal Disruptions:** While external risks are extensively studied, internal disruptions such as dispatching delays, shipment slowdowns, or carrier underperformance remain underrepresented in the literature.

2. **Need for Lightweight, Interpretable Indicators:** Many existing methods rely on complex, data-hungry models that are difficult to deploy in real time with limited information. There is a need for simple, interpretable, and data-efficient approaches that can act as real-time indicators of disruption.

This thesis addresses these gaps by proposing a framework for modeling dispatch and shipment processes through delay-focused indicators. The framework leverages uncertainty intervals and stochastic models to provide delivery time estimations even with limited data. In doing so, it contributes to the broader goals of the M4ESTRO project by enabling real-time detection of internal disruptions and supporting resilient decision making in manufacturing supply chains.

Chapter 3

Dataset

This chapter presents the data that supported the design of a solution addressing the requirements set out in the introduction. Our approach centers on a primary dataset of pilot orders: to enrich this foundational data, we integrated several external services selected based on an assessment of data quality and cost constraints, ensuring compliance with the company's budget. The following sections detail each data source employed.

3.1 Pilot Data

The primary dataset was provided by FAE Technology [55], a company specialized in embedded and custom electronic solutions. While the data originates from a specific industrial context, the nature of the products is irrelevant to this work. The critical aspect of the dataset is its structure, which centers on order records and their associated statuses, ensuring its representativeness of supply chain processes across diverse domains.

Table 3.1 summarizes the main attributes considered in this study, together with their data types and a concise description.

Column	**Type**	**Description**
id	Integer	Unique identifier for each order.
estimatedDeliveryDate	Timestamp	Expected delivery date of the shipment.
distributorId	Integer	Identifier of the supplier.
createdAt	Timestamp	Date and time at which the order was created.
confirmedDeliveryDate	Timestamp	Actual confirmed delivery date.
trackingLink	String	URL to the shipment tracking information.

Table 3.1: Schema of FAE Technology database.

The dataset comprised several thousand records; however, only a small fraction included valid tracking links suitable for retrieving the shipment process steps relevant to our study. New entries with tracking links were inserted

monthly, as reported in Table 3.2. Nevertheless, the number of usable tracking links remained limited throughout the observation period, a circumstance that significantly influenced the methodological approach adopted in this work.

Version	Month	Records	Tracking Links	Tracking Links %
1	Feb 2025	4954	25	0.50
2	Mar 2025	5447	37	0.68
3	May 2025	5855	51	0.87
4	Jul 2025	6551	98	1.50

Table 3.2: Database dump versions and coverage of tracking links.

During the data analysis phase, we identified and addressed the following critical issues:

- **Multiple tracking links per record:** Some entries contained multiple tracking links separated by commas. We resolved this by splitting such records into multiple rows, each retaining the same metadata but assigned a unique identifier.
- **Unreliable estimated delivery dates:** The *estimatedDeliveryDate* field represented an approximate forecast based on historical expectations for each supplier. Comparison with carrier-provided delivery data (see Section 3.2) revealed significant inaccuracies in these estimates. Consequently, we excluded this field from our analysis and relied exclusively on delivery dates provided directly by carriers.

3.2 Tracking Data

Analysis of the tracking links revealed that multiple carriers were employed for order fulfillment. According to information provided by the pilot, the choice of carrier was not determined by the company itself, but by the individual supplier responsible for each order. Examination of the dataset indicated that, in practice, only a limited set of popular carriers was predominantly used, as summarized in Table 3.3.

To keep our approach independent of any specific carrier, we relied on an API broker for shipment tracking data, provided by 17Track [39]. This API offers a unified format for shipment data, regardless of the carrier, enabling consistent integration of tracking information across different logistics providers.

Carrier	N. Orders	N. Orders %
DHL Express	4	4.08
DHL	12	12.24
FedEx	13	13.27
UPS	69	70.41

Table 3.3: Number of orders per carrier for database version 4.

A complete example of the unified format returned by the API is available at [40]. For our purposes, we focus on a subset of this data consisting of a list of objects, each representing a shipment step with a timestamp, a location, and a status, as illustrated in Listing 1. Note that multiple steps may occur at the same location, which is common and corresponds to status updates recorded within the same facility.

Listing 1 Example JSON representing the steps of a shipment.

```
[
    {
        "timestamp": "2025-01-01T20:00:00Z",
        "location": "Gazzaniga, IT",
        "status": "Delivered"
    },
    {
        "timestamp": "2025-01-01T10:00:00Z",
        "location": "Bergamo, IT",
        "status": "InTransit"
    },
    {
        "timestamp": "2025-01-01T05:00:00Z",
        "location": "Milan, IT",
        "status": "inTransit"
    },
    {
        "timestamp": "2025-01-01T00:00:00Z",
        "location": "Milan, IT",
        "status": "PickedUp"
    }
]
```

Analysis of the tracking data further revealed that shipments from the same supplier could originate from multiple locations worldwide, indicating that a single supplier may operate several sites in different regions. To assign a specific location to each supplier site, we relied on the first event reported in the shipment tracking information, based on the assumption (confirmed by the pilot company) that carriers collect orders directly from suppliers.

Examining the carrier usage per site, we observed that most supplier sites consistently used the same carrier, with only one site employing multiple carriers. Nevertheless, for robustness, our modeling approach allows each site to potentially use multiple carriers, ensuring generality and flexibility in handling diverse logistics scenarios.

Site	Supplier	Origin	Orders	%
1	1	Leeds, England, GB	5	5.10
2	2	Louisville, Kentucky, US	3	3.06
3	2	Thief River Falls, Minnesota, US	61	62.24
4	3	Grand Prairie, Texas, US	5	5.10
5	4	Montagnola, Ticino, CH	1	1.02
6	4	Peabody, Massachusetts, US	3	3.06
7	4	Shenzhen, Guangdong, CN	1	1.02
8	5	Dongguan, Guangdong, CN	2	2.04
9	5	Shenzhen, Guangdong, CN	1	1.02
10	7	Hong Kong, Unknown, HK	1	1.02
11	7	Zhuhai, Guangdong, CN	1	1.02
12	8	Warsaw, Mazovia, PL	1	1.02
13	9	Ho Chi Minh City, Ho Chi Minh, VN	1	1.02
14	9	San José, San José, CR	1	1.02
15	10	Bratislava, Bratislava Region, SK	1	1.02
16	10	Gdansk, Pomerania, PL	3	3.06
17	11	Shenzhen, Guangdong, CN	3	3.06
18	12	Boca Raton, Florida, US	1	1.02
19	13	Shenzhen, Guangdong, CN	1	1.02
20	14	Minneapolis, Minnesota, US	1	1.02
21	16	Tsuen Wan, Tsuen Wan, HK	1	1.02

Table 3.4: Overview of all sites with corresponding suppliers and orders.

Table 3.4 presents all identified supplier sites with their corresponding order volumes. Examining the distribution, we observed that the majority of the available orders (61 out of 98) originated from a single site located in Thief River Falls, Minnesota, US. This site represents the only location with a statistically significant number of observations. Although we were aware that such data imbalance would inevitably affect the robustness of our model, we nonetheless aimed to design a formal framework capable of operating on larger datasets, with the expectation of accessing more comprehensive data in future stages of the project.

As a preliminary step, we set aside 10 out of the 98 orders containing tracking links from the main dataset to be used exclusively for model validation. This resulted in two distinct subsets:

- A training set of 88 records, used for model tuning;
- A test set of 10 records, completely excluded from training and reserved solely for evaluation.

The selection of the test set was not entirely random: we chose records from supplier sites with at least five orders to mitigate the impact of the limited dataset. Table 3.5 reports an overview of the orders selected for forming the test set.

Site	Supplier	Orders	Training Orders	Test Orders
1	1	61	53	8
3	2	5	4	1
4	3	5	4	1

Table 3.5: Overview of the selected orders for the test set.

3.3 Geographical Data

With the need for standardized supplier site identification established, the next step was to define a consistent approach to represent the key geographical steps of each shipment. Although we relied on a single API for tracking data, additional processing was required to ensure that locations were uniformly identified and structured. Our approach was designed to satisfy the following requirements:

1. Provide a consistent standard for location identifiers. A canonical representation ensures that the same location can be referenced uniformly across different parts of the application and by external services. This avoids ambiguity (e.g., "NYC" vs. "New York" vs. "JFK") and facilitates seamless integration with third-party systems.

2. Offer an appropriate level of granularity. Locations should not be excessively detailed (e.g., street-level, which introduces noise) nor overly coarse (e.g., country-level, which omits essential logistics information). The chosen granularity must provide meaningful checkpoints for both operational tracking and customer-facing updates.

3. Remain developer-friendly. The format should be straightforward to interpret and apply, reducing the likelihood of errors and easing both implementation and long-term maintenance.

Based on these considerations, the *city-level granularity* turned out as a natural choice to accomplish the previous points. In particular, we adopted the following format:

```
"{CITY}, {STATE}, {COUNTRY}"
```

where all fields are mandatory, and the `STATE` is used to distinguish between cities with the same name within the same country.

This format is readily compatible with regular expressions and corresponds closely to the structure of responses provided by GeoNames [52], the API selected to facilitate the standardization of shipment locations.

3.4 Traffic and Weather Data

As discussed in the introductory Section 1.4, one of the key requirements of the system was the ability to integrate real-time internal supply chain conditions into the computation of estimated travel times and potential delays. In this context, we focused primarily on two factors: road traffic and weather conditions.

To obtain accurate and actionable data, we evaluated multiple services providing real-time traffic and weather information. Ultimately, the services that offered the best combination of data quality, coverage, and alignment with the company's budget constraints were selected:

- **Traffic:** TomTom API [56];
- **Weather:** Visual Crossing API [48].

In the following, we introduce the specific data we chose to utilize for the system.

3.4.1 Traffic Conditions

Upon reviewing the TomTom API documentation, we observed that it offers a wide range of advanced navigation features, many of which exceeded the immediate needs of our system. Our primary objective was to quantify road traffic between a source and destination at a given departure time.

To achieve this, we selected a small set of essential metrics that provide a clear and practical indication of traffic conditions:

1. The estimated travel time **considering real-time traffic**;
2. The estimated travel time **assuming ideal traffic conditions**, i.e., without real-time congestion.

These metrics allow us to calculate preliminary traffic-adjusted transit estimates for specific road sections, providing a foundation for more sophisticated models in subsequent developments.

3.4.2 Weather Conditions

A similar rationale was applied to weather data. The Visual Crossing API provides a comprehensive set of weather metrics, many of which were outside the immediate scope of this initial system implementation. Accordingly, we focused on a limited but highly informative subset of the available data, with the option to expand in future iterations.

Specifically, for a given location and time, we retrieve the following information:

- The real-time weather condition, encoded through a code as a categorical value (e.g., Sun, Rain, Thunderstorm, ...). Details of the codes can be found at [47].
- The real-time temperature.

This approach provides a straightforward and interpretable representation of environmental conditions, allowing the system to adjust travel time estimates and risk assessments accordingly.

3.5 Holiday Data

As discussed in Section 1.4, the pilot company identified supplier holiday closures as one of the primary sources of delays within their supply chain. Accordingly, a key requirement of the system was the integration of holiday-related information, providing an initial method to account for supplier closures when estimating delivery times.

To achieve this, we evaluated external services offering comprehensive worldwide holiday data. Considering both data quality and budget constraints, the most suitable option was found to be Calendarific [42], an API that provides a detailed list of holidays for each country.

Upon analyzing the retrieved data, we observed several categories of holidays, including:

- National, regional, local, or municipal holidays;
- Religious holidays and observances;
- Weekend holidays and working holidays, i.e., specific days that fall on a weekend during which companies remain operational.

Based on these categories, we established the following assumptions regarding supplier site closures:

- Sites are closed on national holidays but remain open for local holidays or observances;
- Sites operate normally during religious holidays and observances, unless the festivity is also classified as national;
- Sites are closed on weekends, except on designated working holidays.

With regard to carriers, once a shipment has been initiated, we assume that holidays do not significantly interrupt the transportation process. This assumption aligns with the operational practices of major international carriers, such as DHL, FedEx, and UPS, which typically maintain continuous transportation networks [36]–[38]. However, it may not strictly hold during major holidays, such as Christmas or New Year's, or for smaller carriers subject to stricter operational constraints.

Chapter 4

Problem Formulation

This chapter establishes the mathematical foundation for modeling and predicting shipment behavior in complex supply chain networks. We present two complementary modeling frameworks that together provide a comprehensive approach to analyzing and predicting supply chain delivery times.

The first framework, *Logistic Chain Modeling*, develops a graph-based representation of supply chain networks using directed acyclic graphs. In this representation, vertices correspond to facilities (e.g., supplier sites, hubs, and manufacturers), and edges represent transportation routes. This foundation enables the use of Markov chains to model routing decisions as stochastic processes with transition probabilities estimated from empirical data.

The second framework introduces the concept of *indicator*, defined as a function that models a specific aspect of the supply chain using quantitative measurements derived from statistical tools. Indicators provide a formal means to capture quantitative aspects of operational performance, making them valuable for decision-making. We distinguish between two types of indicators:

- **Historical indicators:** computed solely from past data, providing baseline distributions for dispatch times, shipment durations, and delivery performance.
- **Realtime indicators:** derived from historical indicators and leveraging the supply chain graph structure to incorporate current factors, such as shipment progress, traffic and weather conditions, into adaptive predictions. They are organized hierarchically, with higher-level realtime indicators aggregating lower-level ones to provide insights at multiple levels of detail.

The proposed methodology is designed to address the inherent uncertainties in supply chain operations: it supports both point estimates, which enable operational planning, and interval-based bounds, which allow for robust decision-making under uncertainty. For both types of indicators, we focus on providing a formal and coherent definition that accommodates multiple possible implementations: we then present our specific solution in Chapter 5.

4.1 Logistic Chain Modeling

In this section, we introduce a formal framework to model shipment behavior in supply chains. Our goal is to represent the logistic network, quantify shipment flows, and model routing decisions using a probabilistic approach. We begin by defining a graph-based structure of the network, introduce flow and Markov models for routing, and then extend the framework to incorporate carrier-specific behavior. Finally, we present the algorithms that enables the efficient execution of the proposed model.

4.1.1 Terminology

Starting from this section, we introduce the following terminology, which will be used throughout the remainder of the text:

- A **supplier site** is also referred to as a *site*;
- An **intermediate location** is also referred to as an *intermediate*;
- An **arc** is also referred to as a *carrier route* (or simply a *route*).

4.1.2 Supply Chain Graph Structure

Definition

Let:

- $C = \{c_1, c_2, \ldots, c_n\}$ denote the set of **carriers**;
- $S = \{s_1, s_2, \ldots, s_m\}$ denote the set of **supplier sites**;
- $I = \{i_1, i_2, \ldots, i_l\}$ denote the set of **intermediate locations**, extracted from the carrier event data associated with the shipments;
- m denote the **manufacturer** and $M = \{m\}$ the singleton containing it.

We define a ***Supply Chain Graph (SCGraph)*** as a **directed graph** $G = (V, A, C)$ with the following properties:

- The set of vertices is a partition $V = S \cup I \cup M$, representing sites, intermediates and the manufacturer, respectively.
- The set of arcs is partition $A = A_{S\to I} \cup A_{I\to I} \cup A_{I\to M}$[1], where each subset corresponds to a specific type of connection:
  - $A_{S\to I} \subseteq S \times I$ denotes the set of arcs from supplier sites to intermediates;
  - $A_{I\to I} \subseteq I \times I$ denotes the set of arcs between intermediates;
  - $A_{I\to M} \subseteq I \times M$ denotes the set of arcs from intermediates to the manufacturer.
- All supplier sites are *source nodes*, i.e., they have zero in-degree: $d_{\text{in}}(s) = 0 \quad \forall s \in S$;

[1] Note that no arc is possible between pairs of sites or from a site to the manufacturer.

- The manufacturer is a *sink node*, i.e., it has zero out-degree: $d_{\text{out}}(m) = 0$.
- The graph is acyclic, meaning it contains no cycles.

A generic representation of a *SCGraph* is shown in Figure 4.1.

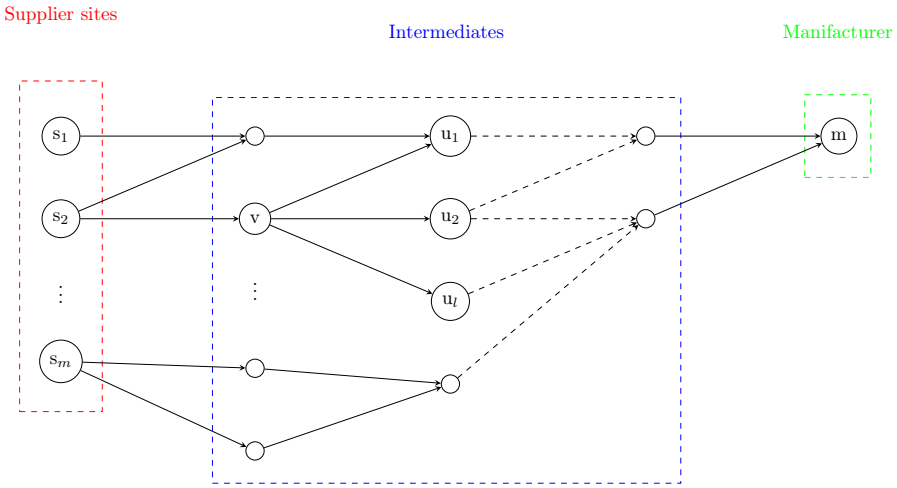


Figure 4.1: Generic representation of a Supply Chain Graph.

We define a *shipment* as a path $\pi = (v_0, v_1, \ldots, v_k = m)$ terminating at the manufacturer vertex. Theorem A.1 shows that, given the specific topology of the graph, any shipment coincides necessarily with a *maximal path* (Definition A.1) ending at the manufacturer vertex m.

4.1.3 Modeling Shipment Flows

Route Flow Function

For any given SCGraph, we define the *route flow function* as

$$f : V^2 \to \mathbb{N}$$

which assigns to each directed arc the number of shipments traversing it, with $f(v, u) = 0$ for any pair $(v, u) \notin A$. This formulation is inspired by the classical *maximum flow problem* [1], although our focus is on discrete shipments and additional constraints specific to supply chains.

Additionally, for any f we impose the following *flow balance constraint* at every intermediate node $i \in I$:

$$\sum_{(i,v)\in A} f(i, v) = \sum_{(v,i)\in A} f(v, i)$$

ensuring that each intermediate node has equal incoming and outgoing flow.

Vertex-Level Flow Function

Leveraging the flow balance constraints, we are able to define the *vertex-level flow function*:

$$f_v(v) = \begin{cases} \sum_{(v,u)\in A} f(v,u), & \text{if } v \in S \cup I, \\ \sum_{(u,v)\in A_{I\to M}} f(u,v), & \text{if } v \in M \end{cases} \qquad f_v : V \to \mathbb{N}$$

representing the total flow from or to a node.

Aggregated Flow

It is possible to further generalize the *flow function*, defining the *aggregated flow* over any subset $U \subseteq V$:

$$f_V(U) = \sum_{u\in U} f_v(u), \qquad f_V : \mathcal{P}(V) \to \mathbb{N}$$

which can be expanded to:

$$f_V(S) = \sum_{s\in S} f_v(s) = \sum_{(s,i)\in A_{S\to I}} f(s,i),$$
$$f_V(M) = f_v(m) = \sum_{(i,m)\in A_{I\to M}} f(i,m)$$

Flow Conservation

Finally, as proved in Theorem A.2, we conclude that the total number of shipments originating from sites, transiting through intermediates, and reaching the manufacturer must be conserved across the entire network. That is:

$$f_V(S) = f_V(M)$$

In other words, the subsets $A_{S\to I}$ and $A_{I\to M}$ constitute an arc cut whose total weight equals the number of shipments in the supply chain network.

This flow conservation principle ensures that every shipment considered in the model corresponds to a complete journey from a supplier site to the manufacturer, without any loss or duplication. This structural property is crucial for the probabilistic modeling developed in the following sections, as it guarantees that all shipment trajectories begin and end at well-defined nodes in the graph.

4.1.4 Probabilistic Routing with Markov Chains

In order to model routing behavior within a supply chain network, we represent shipment trajectories as realizations of a stochastic process. In particular, we employ a discrete-time, time-homogeneous *Markov chain*, where each node corresponds to a facility (e.g., supplier, hub, warehouse), and transitions reflect the **empirical probabilities**[2] of observed routing decisions derived from historical shipment data.

[2]All probabilities are obtained from observed shipment data and thus correspond to *empirical probabilities*, without assuming an underlying theoretical distribution. For ease of reading, in the following sections we will nevertheless refer to them simply as *probabilities*.

Assumptions

This modeling approach relies on the following assumptions:

- **Markov property**: Routing decisions at each node depend only on the current location, independent of the full past trajectory [7].
- **Stationarity**: The transition probabilities estimated from historical data are stable over time [11].

Transition Probabilities

Let $f : V^2 \to \mathbb{N}$ denote the number of shipments from node v to node u, and $f_v : V \to \mathbb{N}$ the total shipments departing from v.

The *transition probability* from node v to node u is defined as:

$$P_{vu} = \mathbb{P}(X_{t+1} = u \mid X_t = v) = \frac{f(v,u)}{f_v(v)}, \quad \forall (v,u) \in A$$

which constructs a time-homogeneous Markov chain $\{X_t\}_{t\in\mathbb{N}}$ over state space V. To ensure that the transition matrix $P = [P_{vu}]$ is a valid row-stochastic matrix satisfying

$$\sum_{u\in V} P_{vu} = 1, \quad \forall v \in V$$

we impose the additional condition that the sink vertex m is indeed absorbing by setting

$$P_{mm} = \mathbb{P}(X_{t+1} = m \mid X_t = m) = 1$$

This condition reflects the fact that reaching the absorbing state m corresponds to the successful delivery of a shipment to the manufacturer.

Path Probabilities

By construction of the SCGraph, all maximal paths in the graph terminate at m (see Theorem A.1). Once a shipment reaches m, it remains there indefinitely, and every other node $v \in V \setminus \{m\}$ lies on a path that eventually leads to m. This guarantees that every realization of the Markov process is absorbed at m in finite time with probability 1.

Consider a maximal path (complete shipment trajectory)

$$\pi = (v_0, v_1, \ldots, v_k = m)$$

The probability of observing this particular path under the Markov model is:

$$\begin{aligned}\mathbb{P}(\pi) &= \mathbb{P}(X_0 = v_0, X_1 = v_1, \ldots, X_k = m) \\ &= \mathbb{P}(X_0 = v_0) \cdot \prod_{i=0}^{k-1} \mathbb{P}(X_{i+1} = v_{i+1} \mid X_i = v_i).\end{aligned}$$

If the initial node $v_0 \in S$ is fixed (e.g., known or externally selected), then the conditional probability of the path simplifies to:

$$\mathbb{P}(\pi \mid X_0 = v_0) = \prod_{i=0}^{k-1} P_{v_i v_{i+1}} = \prod_{i=0}^{k-1} \frac{f(v_i, v_{i+1})}{f_v(v_i)}$$

Hence, the likelihood of any shipment path corresponds to the product of the transition probabilities along the path in the Markov chain induced by historical shipment flows. The unique absorbing state m, structurally guaranteed by the SCGraph, ensures that all shipment processes ultimately terminate at this final destination.

Path Distributions Probability

Given a vertex $v \in V$, let $\Pi(v) = \{\pi_1, \pi_2, \ldots, \pi_\ell\}$ denote the set of all maximal paths originating from v, where each path

$$\pi_i = (v_0 = v, v_1, \ldots, v_{k_i} = m), \quad \forall i = 1, \ldots, \ell$$

terminates at the absorbing node m. We define the associated *probability distribution* $\boldsymbol{p} = (p_1, p_2, \ldots, p_\ell)$, where each entry represents the probability of observing the corresponding path under the Markov model:

$$p_i = \mathbb{P}(\pi_i \mid X_0 = v), \quad \forall i = 1, \ldots, \ell$$

The Theorem A.3 shows how this vector satisfies the standard probability constraints:

- $p_i \in [0, 1] \quad \forall i = 1, \ldots, \ell$;
- $\sum_{i=1}^{k} p_i = 1$.

The probability distribution $\boldsymbol{p}$ describes the likelihood all the possible shipment paths from the starting point v to the final destination m, under the routing model.

4.1.5 Carrier-Conditioned Routing Dynamics

Analyzing the data retrieved from carrier shipments, we noticed that different carriers could potentially follow different routes from a same starting location. As a consequence, we decided that the routing behavior within the supply chain must depend also on the specific *carrier* executing the shipment. To capture this, we define transition probabilities *conditioned* on the carrier $c \in C$. This allows us to model distinct routing patterns and preferences for each logistic provider.

Carrier-Conditioned Flow Functions

Let $f^c : V^2 \to \mathbb{N}$ denote the route flow function restricted to shipments handled by the carrier c:

$$f^c(v, u) = \text{number of shipments from } v \text{ to } u \text{ executed by carrier } c.$$

Accordingly, define the carrier-specific vertex-level flow:

$$f_v^c(v) = \sum_{(v,u) \in A} f^c(v, u).$$

and the carrier-specific aggregated-level flow

$$f_V^c(U) = \sum_{u \in U} f_v^c(u).$$

Carrier-Conditioned Transition Probabilities

Given these definitions, the transition probability from v to u conditioned on carrier c is:

$$P^c_{vu} = \mathbb{P}(X_{t+1} = u \mid X_t = v, C = c) = \frac{f^c(v,u)}{f^c_v(v)}, \qquad \forall (v,u) \in A \text{ s.t. } f^c_v(v) > 0$$

As with the general case, to ensure that the carrier-conditioned transition matrix $P^c = [P^c_{vu}]$ is row-stochastic for each $c \in C$, we impose:

$$\sum_{u \in V} P^c_{vu} = 1 \quad \forall v \in V \text{ s.t. } f^c_v(v) > 0$$

and, for the absorbing node m, we explicitly set:

$$P^c_{mm} = 1, \quad \forall c \in C$$

Moreover, for any node $v \in V$ where no shipments with carrier c were observed to depart (i.e., $f^c_v(v) = 0$), we define a degenerate self-loop:

$$P^c_{vv} = 1$$

This guarantees that each row of the transition matrix sums to 1, even in the absence of observed transitions for certain nodes under a given carrier.

Carrier-Conditioned Path Probabilities

Let $\pi = (v_0, v_1, \ldots, v_k = m)$ be a *maximal path*. The probability of this path under the Markov model for carrier c, given the shipment starts at v_0, is:

$$\mathbb{P}(\pi \mid X_0 = v_0, C = c) = \prod_{i=0}^{k-1} P^c_{v_i v_{i+1}} = \prod_{i=0}^{k-1} \frac{f^c(v_i, v_{i+1})}{f^c_v(v_i)} \tag{4.1}$$

This formulation enables a more fine-grained analysis of the shipment dynamics across different carriers, accounting for heterogeneous behavior in network traversal and routing strategies.

Carrier-Conditioned Path Distributions Probability

Similarly, for each carrier $c \in C$, and denoting by $\Pi(v) = \{\pi_1, \pi_2, \ldots, \pi_\ell\}$ the set of all maximal paths from a vertex v to m, we define the *carrier-conditioned path distribution vector*:

$$\boldsymbol{p}^c = (p^c_1, p^c_2, \ldots, p^c_\ell)$$

where each component represents the probability of observing a given path $\pi_i = (v_0 = v, v_1, \ldots, v_{k_i} = m) \in \Pi(v)$ under the routing behavior of carrier c:

$$p^c_i = \mathbb{P}(\pi_i \mid X_0 = v, C = c) = \prod_{j=0}^{k_i - 1} P^c_{v_j v_{j+1}} \tag{4.2}$$

The vector $\boldsymbol{p}^c$ satisfies the same probabilistic constraints:

- $p^c_i \in [0,1] \quad \forall i = 1, \ldots, \ell$;

- $\sum_{i=1}^{\ell} p_i^c = 1$

This allows us to model the distribution of paths specific to each carrier, reflecting their distinct logistical strategies and network usage.

4.1.6 Supply Chain Graph Construction

The Supply Chain Graph (SCGraph) is derived directly from the pilot orders (Section 3.1) and the associated tracking information (Section 3.2).

Given a delivered order

$$o = (s, c, m)$$

with supplier site s, carrier c, and manufacturer m, and the sequence of tracking steps

$$ts = (ts_1, ts_2, \ldots, ts_k = m)$$

the graph is updated as follows:

1. **Supplier site identification.** Select the supplier site vertex corresponding to ts_1. If the vertex does not exist, create a new supplier site node.
2. **Arc insertion and flow update.** For each consecutive pair of steps $(ts_j, ts_{j+1}) \in ts$:
   - If the corresponding arc is absent in the graph, create it and initialize its carrier-specific flow as $f^c(ts_j, ts_{j+1}) = 0$, ignoring eventual self loops.
   - Increment the flow $f^c(ts_j, ts_{j+1})$ by one unit.
3. **Transition probability update.** Update the transition probability matrix P^c_{vu} to reflect the revised flow counts, according to Equation (4.1).

This procedure incrementally maintains both the structural and probabilistic properties of the supply chain graph as new shipment data is ingested.

Starting from the available data, we applied the procedure described above to construct the supply chain graph. Its main characteristics are summarized in Table 4.1.

A visualization of the SCGraph is provided in Figure 4.2. The following conventions were adopted for readability:

- **Vertex color** denotes the partition:
  - Supplier sites;
  - Intermediates;
  - Manufacturer.
- **Vertex and arc size** is proportional to the normalized shipment flow associated with the corresponding node or edge.
- Supplier site vertices are labeled with their site identifiers. For intermediates, only the country code is reported to avoid clutter.

Metric	Description	Value	Share
$\|V\|$	Number of vertices	66	
$\|S\|$	Supplier sites	21	31.82%
$\|I\|$	Intermediates	44	66.67%
$\|M\|$	Manufacturers	1	1.52%
$\|A\|$	Number of arcs	82	
$\|A_{S\to I}\|$	Supplier → Intermediate arcs	24	29.27%
$\|A_{I\to I}\|$	Intermediate → Intermediate arcs	55	67.07%
$\|A_{I\to M}\|$	Intermediate → Manufacturer arcs	3	3.66%

Table 4.1: Supply Chain Graph characteristics.

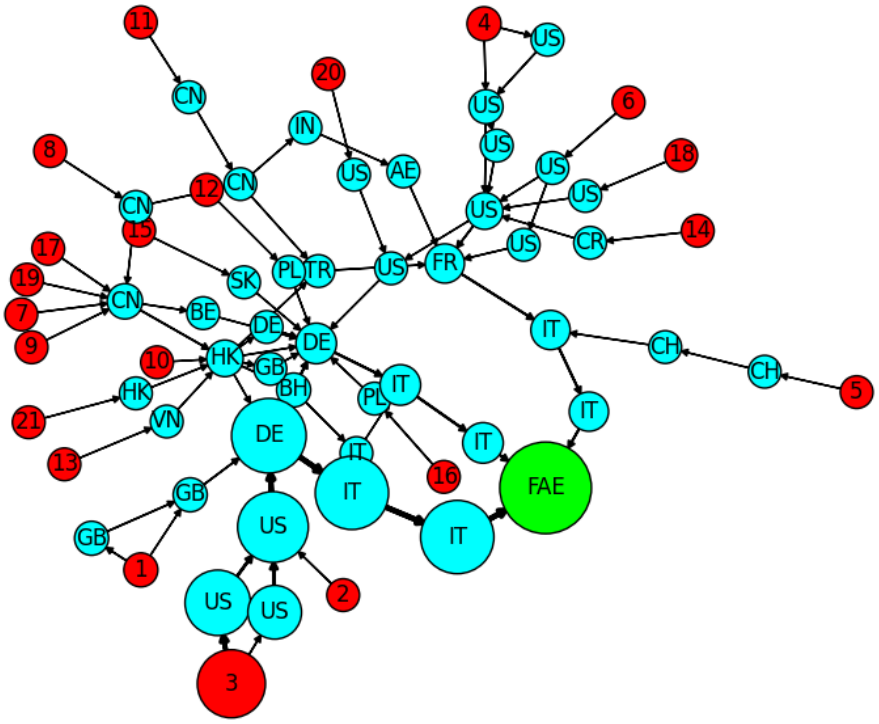


Figure 4.2: Supply Chain Graph visualization.

Cycle Detection and Resolution

To enforce the acyclicity constraint of the SCGraph, we introduced a two-step cycle detection and resolution procedure after step 1 of the graph update process:

1. **Tracking steps cycle resolution.** For a newly concluded shipment ts, we first construct a temporary graph consisting only of the tracking steps in ts. Cycle detection is then performed on this temporal graph using a depth-first search (DFS). If a cycle is detected, for example

$$cycle = (ts_i, ts_{i+1}, \ldots, ts_{i+j}, ts_i)$$

the entire cycle is collapsed into a single repeating vertex, removing the intermediate steps $ts_{i+1}, \ldots, ts_{i+j}$, and the procedure is restarted on the updated graph. This ensures that the sequence of tracking steps ts is acyclical before integration. More sophisticated detection algorithms, such as Tarjan's strongly connected components algorithm [2], could also be employed to detect all cycles simultaneously.

2. **Integration and global cycle verification.** Once ts is acyclical, it is integrated into the main SCGraph. A new cycle detection is then performed on the updated graph. If integration introduces a cycle, the corresponding order is discarded to preserve the acyclicity of the SCGraph.

This two-step procedure consolidates cycle travel times within individual shipments while preventing cycles at the global graph level, maintaining the consistency of the supply chain model. It is worth noting that Step 1 may alter the residence time of the repeating vertex, potentially creating outliers in the data. In this preliminary implementation, we rely on the fact that cycles are expected to be extremely rare (as confirmed by our dataset below) and handle these cases with standard outlier detection methods. If the frequency of detected cycles increases, alternative strategies should be evaluated.

Cycle Instances in the Dataset. In our dataset, a single cycle instance was observed, corresponding to a back-and-forth travel between:

- Leipzig, Saxony, DE
- Paris, Île-de-France, FR

The shipment had the form

$$ts = (s, \ldots, \text{Leipzig}, \text{Paris}, \text{Leipzig}, \ldots, m)$$

which we interpreted as a carrier error. Using the two-step procedure, Step 1 alone was sufficient to resolve this cycle by collapsing the repeated Leipzig-Paris steps and producing the new sequence

$$ts' = (s, \ldots, \text{Leipzig}, \ldots, m)$$

4.1.7 Paths Extraction

Given the particular structure of the SCGraph, we can exploit the fact that all maximal paths necessarily terminate at the manufacturer (Theorem A.1) in order to extract all the possible paths from a given vertex.

Let $v \in V$ be a vertex: then we can compute the set of maximal paths from v to m

$$\Pi(v) = \{\pi_1, \pi_2, \ldots, \pi_\ell\}$$

by simply concatenating all the possible paths from all the neighbours of vertex v. Formally:

$$\text{paths}(v) = \begin{cases} \{(m)\} & \text{if } v = m, \\ \bigcup_{(v,u)\in A} \; \bigcup_{\pi \in \text{paths}(u)} \text{prepend}(v, \pi) & \text{otherwise} \end{cases}$$

where

$$\text{prepend}\big(v,\, \pi = (v_0, v_1, \ldots, v_k)\big) = (v, v_0, v_1, \ldots, v_k)$$

Algorithmically, we perform a modified **Depth-First Search (DFS)** starting from a vertex v, recursively collecting all paths from its neighbors and prepending the current vertex at each step. Pseudocode for this procedure is given in Algorithm 1: before invoking the procedure, we initialize $color[v] = 0$ for all $v \in V$.

Notice that if the condition on line 9 evaluates to false, the vertex has already been visited ($color[v] = 2$), as the graph is acyclic by construction and $color[v] = 1$ would imply the presence of a cycle. Consequently, after line 11 the set Π_u is guaranteed to be available.

Algorithm 1 Collect all Paths to Manufacturer m

```
1:  procedure DFS(v)
2:      color[v] ← 1
3:      if v = m then
4:          color[v] ← 2
5:          return {(m)}
6:      end if
7:      Π_v ← ∅
8:      for all (v, u) ∈ A do
9:          if color[u] = 0 then
10:             Π_u ← DFS(u)
11:         end if
12:         for all π_u ∈ Π_u do
13:             π_v ← prepend(v, π_u)
14:             Π_v ← Π_v ∪ {π_v}
15:         end for
16:     end for
17:     color[v] ← 2
18:     return Π_v
19: end procedure
```

While the algorithm could theoretically lead to exponential time complexity, in practice we do not expect path extraction to constitute a computational bottleneck for our model.[3]

4.1.8 Path Probability Distribution Calculation

Once the set of paths $\Pi(v)$ is determined, the probability associated with each path $\pi \in \Pi(v)$ is computed using the carrier-conditioned transition probabili-

[3] A standard DFS traversal has time complexity $O(|V| + |A|)$. In our case, the `prepend` operation introduces additional cost, potentially yielding an exponential number of paths in theory. However, our dataset shows that carriers follow a limited number of routes (on average 4 paths per vertex, maximum 24), so it is reasonable to bound the number of paths per vertex by $O(|V|)$. Under this assumption, the worst-case complexity is $O(|V|^2 \cdot |A|)$. In practice, only a subset of nodes and arcs is explored, and efficiency can be improved by precomputing paths via dynamic programming, reducing subsequent queries to $O(|V|)$ time.

ties P^c_{vu}:

$$\mathbb{P}(\pi \mid X_0 = v, C = c) = \prod_{(x,y)\in\pi} P^c_{xy}.$$

The evaluation of all path probabilities requires $O(\ell \cdot |V|)$ time, where $\ell = |\Pi(v)|$. In practice, however, we expect path lengths to be considerably smaller than $|V|$, ensuring that the computation remains efficient.[4]

Additional details on the data structures used to support these operations are provided in Appendix B.1.

4.2 Historical Indicators

In this section, we introduce the *historical indicators*, defined as statistical measures that characterize key operational times in the order fulfillment process, based exclusively on historical data. We distinguish three main components:

- the time from order acceptance to dispatch (*dispatch time*);
- the time from dispatch to arrival at the destination (*shipment time*);
- the total time from order acceptance to delivery (*delivery time*).

The first two components are modeled as probability distributions, capturing both their expected values and the associated variability, while the third is obtained as their sum.

An overview of the defined historical indicators is provided in Table B.1.

4.2.1 Dispatch Indicators

For each supplier, we model the statistical distribution of order dispatch times using historical data. Since a single supplier may operate multiple sites across different geographic locations, each site is assigned its own dispatch time distribution.

Formally, the Dispatch Time (DT) is defined as a mapping from supplier sites to probability distributions over non-negative dispatch times:

$$\mathrm{DT} : S \to \mathcal{D}(\mathbb{R}_+)$$

where:

- $\mathcal{D}(X)$ represents the set of probability distributions defined over the domain X;
- $\mathbb{R}_+$ denotes the set of non-negative real numbers:

$$\mathbb{R}_+ = \{x \in \mathbb{R} \mid \ x \geq 0\}$$

[4]Empirical data indicate an average path length of 5.76 and a maximum of 9 for site 4, which supports the conservative bound $\ell = O(|V|)$, yielding a worst-case complexity of $O(|V|^2)$. In practice, only a subset of nodes is explored, and evaluation can be terminated early if the cumulative probability product reaches zero, as occurs for most paths in our dataset. Efficiency can be further improved by precomputing path probabilities and storing the results, so that subsequent queries can be answered in $O(|V|)$ time.

For a given supplier site $s \in S$, this distribution can be summarised in two complementary ways:

- **Pointwise** DT:

$$\mathrm{DT}_\mu(s) = \mathbb{E}[\mathrm{DT}(s)], \quad \mathrm{DT}_\mu : S \to \mathbb{R}_+$$

  This returns the expected dispatch time for site s, providing a single deterministic estimate derived from past records.

- **Interval-based** DT:

$$\mathrm{DT}_\beta(s) = \mathtt{CI}_\beta(\mathrm{DT}(s)) = \left[L_\beta^{disp}(s),\ U_\beta^{disp}(s)\right],$$
$$\mathrm{DT}_\beta : S \to \mathbb{R}_+^2$$

  where $\mathtt{CI}_\beta(\mathrm{DT}(s))$ denotes the two-sided confidence interval at level β, such that:

$$\mathbb{P}\left(L_\beta^{disp}(s) \le X \le U_\beta^{disp}(s)\right) \ge \beta, \quad X \sim \mathrm{DT}(s)$$

  This form explicitly quantifies uncertainty, making it more appropriate for robust planning under variable dispatch conditions.

4.2.2 Shipment Indicators

For each shipment, we model the statistical distribution of transit times using historical records. Unlike dispatch times, shipment times depend jointly on the supplier site of origin and the carrier responsible for transportation. Consequently, each *(site, carrier)* pair is assigned its own shipment time distribution.

Formally, the Shipment Time (ST) is defined as a mapping from supplier site–carrier pairs to probability distributions over non-negative shipment times:

$$\mathrm{ST} : S \times C \to \mathcal{D}(\mathbb{R}_+)$$

For a given pair $(s, c) \in S \times C$, this distribution can be summarised in two complementary ways:

- **Pointwise** ST:

$$\mathrm{ST}_\mu(s, c) = \mathbb{E}[\mathrm{ST}(s, c)], \quad \mathrm{ST}_\mu : S \times C \to \mathbb{R}_+$$

  This returns the expected shipment time for the given *(site, carrier)* pair, providing a single deterministic estimate derived from historical transit data.

- **Interval-based** ST:

$$\mathrm{ST}_\beta(s, c) = \mathtt{CI}_\beta(\mathrm{ST}(s, c)) = \left[L_\beta^{ship}(s, c),\ U_\beta^{ship}(s, c)\right],$$
$$\mathrm{ST}_\beta : S \times C \to \mathbb{R}_+^2$$

  where $\mathtt{CI}_\beta(\mathrm{ST}(s, c))$ denotes the two-sided confidence interval at level β, such that:

$$\mathbb{P}\left(L_\beta^{ship}(s, c) \le X \le U_\beta^{ship}(s, c)\right) \ge \beta, \quad X \sim \mathrm{ST}(s, c)$$

  This form explicitly quantifies uncertainty, making it suitable for robust planning in the presence of variable transit conditions.

4.2.3 Delivery Indicators

The delivery time represents the total elapsed time from order acceptance to final receipt by the customer. For modeling purposes, we make the following simplifying assumption:

Assumption. For any supplier site $s \in S$ and carrier $c \in C$, the dispatch time $\mathrm{DT}(s)$ and the shipment time $\mathrm{ST}(s, c)$ are treated as independent.

The raw data from supplier site 3 and carrier UPS, the only site-carrier pair with enough observations, support this assumption, with a correlation coefficient of 0.043 between dispatch and shipment times, indicating no linear dependence.

Based on this independence, pointwise and interval-based Delivery Time (DLT) indicators are computed as simple sums of the corresponding dispatch and shipment metrics:

- **Pointwise** DLT: the expected delivery time for pair (s, c) is the sum of expectations:

$$\mathrm{DLT}_\mu(s, c) = \mathrm{DT}_\mu(s) + \mathrm{ST}_\mu(s, c), \quad \mathrm{DLT}_\mu : S \times C \to \mathbb{R}_+$$

- **Interval-based** DLT: for practical interpretability and computational simplicity, we construct delivery intervals by summing the corresponding dispatch and shipment interval endpoints:

$$\mathrm{DLT}_\beta(s, c) = \left[L_\beta^{\mathrm{del}}(s, c),\, U_\beta^{\mathrm{del}}(s, c)\right]$$

with

$$L_\beta^{\mathrm{del}}(s, c) = L_\beta^{\mathrm{disp}}(s) + L_\beta^{\mathrm{ship}}(s, c), \qquad U_\beta^{\mathrm{del}}(s, c) = U_\beta^{\mathrm{disp}}(s) + U_\beta^{\mathrm{ship}}(s, c)$$

While this approach does not preserve the exact confidence level β for the combined interval, it provides a conservative and interpretable approximation suitable for operational decision-making.

4.3 Realtime Indicators

In this section, we introduce the **realtime indicators**, designed to quantify measurable metrics for predicting shipment behavior using both historical and real-time data. These indicators leverage the *Supply Chain Graph* model as well as the *historical indicators* defined in previous sections.

The indicators are structured across multiple levels of granularity, ranging from individual network components to system-wide aggregations. Higher-level indicators synthesize information from lower levels, offering progressively comprehensive insights into the supply chain's state.

A concise overview of all the defined real-time indicators is provided in Table B.2.

4.3.1 Definitions

Time Sequence

We define the following time variables:

- $t_0 \in \mathbb{R}_+$: the time at which the order is placed (*order time*);
- $t_e^d \in \mathbb{R}_+$: the timestamp of the most recently registered dispatch event (*dispatch event time*);
- $t^d \in \mathbb{R}_+$: the time at which the dispatch estimation is performed (*dispatch estimation time*);
- $t_1 \in \mathbb{R}_+$: the time the shipment begins, i.e., when the carrier picks up the order at the supplier site (*shipment time*);
- $t_e^s \in \mathbb{R}_+$: the timestamp of the most recently registered shipment event (*shipment event time*);
- $t^s \in \mathbb{R}_+$: the time at which the shipment estimation is performed (*shipment estimation time*);
- $t_n \in \mathbb{R}_+$: the time the order is delivered to the manufacturer (*delivery time*).

Although distinguishing between event-retrieval and estimation times increases complexity, maintaining separate time points for both dispatch and shipment enables asynchronous event retrieval. This ensures correct estimations even when events are not retrieved in real time and simplifies the integration with external event data sources within the broader system.

We further define:

- a *time sequence* as a tuple

  $$\boldsymbol{t} = (t_0, t_e^d, t^d, t_1, t_e^s, t^s, t_n)$$

  respecting the constraint

  $$t_0 \leq t_e^d \leq t^d \leq t_1 \leq t_e^s \leq t^s \leq t_n$$

- the set of *possible times* as the closed interval

  $$T = [t_0, t_n] \subset \mathbb{R}$$

  which can be partitioned as follows:

  - The set of *possible dispatch times*:

    $$T^d = [t_0, t_1]$$

  - The set of *possible shipment times*:

    $$T^s = [t_1, t_n]$$

    For convenience in modeling, we also define the *shifted shipment times*:

    $$T_0^s = [0, t_n - t_1]$$

    as the shipment times measured relative to the start of the shipment interval.

Operational Stages

Two distinct operational stages are considered:

- **Dispatch Stage**: the shipment has not yet started, i.e., t_1 is unknown and must be estimated. In this case, t_e^s and t^s coincide with t_1 (see Figure 4.3), resulting in the constrained sequence:
  - $t_0 \leq t_e^d \leq t^d \leq t_1 = t_e^s = t^s \leq t_n$.
- **Shipment Stage**: the shipment has already started, i.e., t_1 is known. Here, t_e^d and t^d both coincide with t_1, and the sequence (see Figure 4.4) reduces to:
  - $t_0 \leq t_1 = t_e^d = t^d \leq t_e^s \leq t^s \leq t_n$.

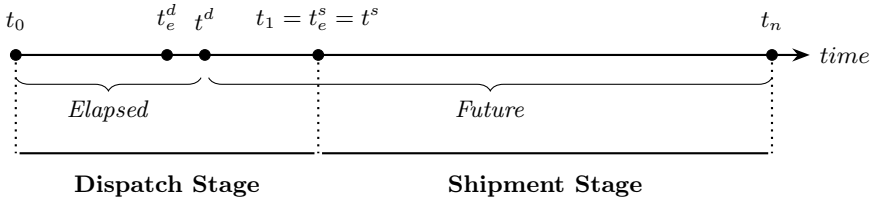


Figure 4.3: Time sequence for the dispatch stage.

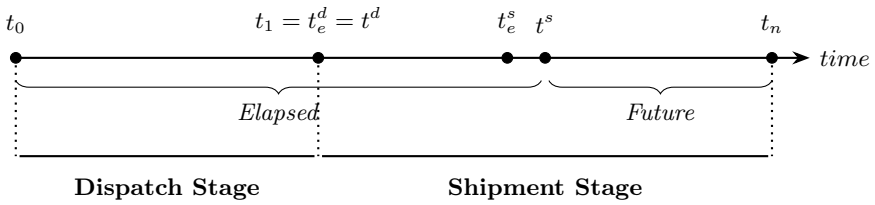


Figure 4.4: Time sequence for the shipment stage.

4.3.2 Vertex Level Indicators

Estimated Dispatch Time

For a given supplier site s, we define the Estimated Dispatch Time (EDT) as the time elapsed between the *order time* t_0 and the *shipment time* t_1:

$$\mathrm{EDT}_\mu(t_0, t_1) = t_1 - t_0 \tag{4.3}$$

The computation of this metric depends on the particular operational stage.

Shipment Stage. In the *shipment stage*, t_1 is known, and the EDT can be computed directly from Equation (4.3), obtaining an instance of valid time sequence reported in Figure 4.4.

Dispatch Stage. In the *dispatch stage*, t_1 must be estimated. To this end, we use the DT indicators introduced in Section 4.2.1. Since our goal is to obtain a single time estimate, the *pointwise* DT indicator is more suitable than its *interval-based* counterpart. Accordingly, we compute a first estimation of the *shipment time* as:

$$t_1' = t_0 + \mathrm{DT}_\mu(s) = t_0 + \mathbb{E}\left[\mathrm{DT}(s)\right]$$

However, a purely pointwise estimate may fail to capture dispatch delays. In particular, both the *dispatch event time* t_e^d and the *dispatch estimation time* t^d can occur after the estimated t_1' if a delay pushes the actual dispatch beyond the expected value of the distribution. To handle this, we adjust the estimate as:

$$t_1 = \max\left(t_1', t^d\right)$$

producing a valid *time sequence*:

$$\mathbf{t} = (t_0, t_e^d, t^d, t_1, t_e^s, t^s, t_n)$$

which satisfies the constraint in Inequality (4.3.1), with t_n remaining the only unknown.

Furthermore, at this point we can estimate the remaining dispatch time at t^d as the difference $t_1 - t^d$.

Holidays. According to the assumptions outlined in Section 3.5, in this model we consider only national and public holidays for supplier sites, assuming that the supplier remains closed for the entire day during a holiday.

To account for downtime due to holidays in the EDT indicator for the *dispatch stage*, we consider all holidays within the dispatch interval T^d, effectively adding one additional day for each holiday encountered.

Algorithm 2 presents the pseudocode for this adjustment, where the function `is_holiday` returns `true` if the specified date is a holiday in the country of the supplier site s, and `false` otherwise.

Overall Residence Index

For each intermediate $i \in I$ representing a facility, we aim to model the distribution of processing times required for orders at that location. This distribution captures the variability in how long orders remain at a facility before proceeding to the next stage of the workflow.

We define the Overall Residence Index (ORI) as a function

$$\mathrm{ORI} : I \to \mathcal{D}(\mathbb{R}_+)$$

that maps each intermediate i to a probability distribution over non-negative residence times, estimated from historical data.

Unlike in the routing analysis, the ORI aggregates processing times across all carriers without distinction. While it is reasonable to assume that carriers have similar processing times in the same locations, this assumption should be empirically verified once more data become available. If significant differences are observed, processing times should be differentiated for each carrier at each location.

Algorithm 2 Estimate Dispatch Time Adjusted for Holidays

Require: Order time t_0, Dispatch estimation time t^d, Supplier site s
Ensure: Adjusted shipment time t_1 accounting for holidays

$days\_remaining \leftarrow \mathrm{DT}_\mu(s)$ ▷ Can be fractional
$days\_elapsed \leftarrow 0$
$current\_date \leftarrow t_0$
while $days\_remaining > 0$ **do**
  $increment \leftarrow \min(1, days\_remaining)$
  $days\_elapsed \leftarrow days\_elapsed + increment$
  if not `is_holiday`($current\_date$, s) **then**
    $days\_remaining \leftarrow days\_remaining - increment$
  end if
  if $days\_remaining > 0$ **then**
    $current\_date \leftarrow current\_date + 1$
  end if
end while
$t_1' \leftarrow t_0 + days\_elapsed$
$t_1 \leftarrow \max\left(t_1', t^d\right)$ ▷ Ensure t_1 reflects any dispatch delay
return t_1

Vertex Time

In order to incorporate the ORI into our modeling framework, we define the Vertex Time (VT) function under two different variants, each reflecting a different level of temporal granularity:

- *Pointwise* VT:

$$\mathrm{VT}_\mu(i) = \mathbb{E}[\mathrm{ORI}(i)], \quad \mathrm{VT}_\mu : I \to \mathbb{R}_+$$

  This function maps each intermediate node $i \in I$ to its expected residence time, offering a scalar estimate suitable for deterministic computations or baseline planning.

- *Interval-based* VT:

$$\mathrm{VT}_\beta(i) = \mathtt{CI}_\beta(\mathrm{ORI}(i)) = \left[L_\beta^v(i),\ U_\beta^v(i)\right], \quad \mathrm{VT}_\beta : I \to \mathbb{R}_+^2$$

  where $\mathtt{CI}_\beta(\mathrm{ORI}(i))$ denotes a two-sided confidence interval at level β, defined such that:

$$\mathbb{P}\left(L_\beta^v(i) \le X \le U_\beta^v(i)\right) \ge \beta \quad \text{for } X \sim \mathrm{ORI}(i)$$

  This variant provides a probabilistic interval estimate that captures the uncertainty inherent in residence times. It is particularly for scenarios where pointwise estimates may be too unreliable or insufficiently robust.

4.3.3 Route Level Indicators

Overall Transit Index

For each route $(v, u) \in A$ connecting two locations in the network, we aim to model the distribution of travel times required for orders to move between

them. This distribution reflects the variability in transit durations, excluding any delays due to processing or handling at intermediate facilities.

We define the Overall Transit Index (OTI) as a function

$$\mathrm{OTI} : V^2 \to \mathcal{D}(\mathbb{R}_+)$$

that maps each route (v, u) to a probability distribution over non-negative transit times, derived from historical shipment data.

Consistent with the modeling approach used for the ORI, the OTI aggregates transit times across all carriers without distinction. This is justified by empirical data indicating that carriers exhibit similar transit times on the same routes. Nevertheless, also this assumption should be re-evaluated as additional data become available.

Traffic Meta Index

Traffic is a crucial factor in modeling shipment time estimations. To enhance the accuracy of our predictions, we incorporate real-time traffic data directly into our model. Specifically, we define the Traffic Meta Index (TMI) as a function

$$\mathrm{TMI} : V^2 \times T \to [0, 1]$$

which assigns a traffic index value to every route for a given time, normalized within the interval $[0, 1]$. This index can be interpreted as follows:

- TMI $= 0$: no traffic congestion on the route;
- TMI $= 1$: extreme traffic congestion on the route.

By including TMI in the estimation process, the model can dynamically adjust expected shipment times based on historical and real-time traffic conditions.

Weather Meta Index

Similarly to the TMI, we incorporate weather conditions into our model through the Weather Meta Index (WMI), defined as

$$\mathrm{WMI} : V^2 \times T \to [0, 1]$$

which assigns to each route a normalized weather severity value for a given time, where:

- WMI $= 0$: ideal weather conditions with no adverse effects on transportation;
- WMI $= 1$: severe weather conditions on the route, significantly impacting travel time.

As for TMI, integrating WMI allows the model to account for delays and uncertainties caused by weather, further refining shipment time predictions.

Route Time

Following the methodology introduced in Section 4.3.2 for defining the VT, we introduce the Route Time (RT), a metric that associates a travel time estimate to each route $(v, u) \in A$. We distinguish between two variants:

- *Pointwise* RT:

$$\mathrm{RT}_\mu(v,u) = \mathbb{E}[\mathrm{OTI}(v,u)], \quad \mathrm{RT}_\mu : V^2 \to \mathbb{R}_+$$

  This provides a single scalar estimate representing the expected travel time for the route.

- *Interval-based* RT:

$$\mathrm{RT}_\beta(v,u) = \mathtt{CI}_\beta(\mathrm{OTI}(v,u)) = \left[L^r_\beta(v,u),\ U^r_\beta(v,u)\right],$$
$$\mathrm{RT}_\beta : V^2 \to \mathbb{R}^2_+$$

  where $\mathtt{CI}_\beta(\mathrm{OTI}(v,u))$ is the two-sided confidence interval at level β, defined such that:

$$\mathbb{P}\left(L^r_\beta(v,u) \le X \le U^r_\beta(v,u)\right) \ge \beta, \quad X \sim \mathrm{OTI}(v,u)$$

  This version captures uncertainty by providing a probabilistic range rather than a single value.

Dynamic Route Time

To enhance the accuracy of route time estimates, we introduce the Dynamic Route Time (DRT), which integrates historical travel time data with real-time traffic and weather information. This approach provides route time estimates that adapt dynamically to current conditions while retaining the statistical grounding of historical observations.

We define two variants of the DRT:

- *Pointwise* DRT:

$$\mathrm{DRT}_\mu(v,u,t) = f\Big(\mathrm{RT}_\mu(v,u), \mathrm{TMI}(v,u,t), \mathrm{WMI}(v,u,t)\Big),$$
$$\mathrm{DRT}_\mu : V^2 \times T \to \mathbb{R}$$

  providing a correction of the expected RT value using traffic and weather information.

- *Interval-based* DRT:

$$\mathrm{DRT}_\beta(v,u,t) = f\Big(\mathrm{RT}_\beta(v,u), \mathrm{TMI}(v,u,t), \mathrm{WMI}(v,u,t)\Big),$$
$$\mathrm{DRT}_\beta : V^2 \times T \to \mathbb{R}^2$$

  which adjusts the confidence interval with the additional data, potentially losing the strict probabilistic interpretation.

4.3.4 Path Level Indicators

In this section, we introduce indicators designed to operate at the path level of granularity. Specifically, we distinguish between two types: the first leverages historical shipment time distributions, while the second exploits the structural characteristics of the underlying graph. Finally, we integrate both approaches into a unified indicator that provides an estimate of the time required to traverse a given path.

Transit Time

We define an indicator that estimates the remaining duration of a shipment based on the historical shipment indicator defined in Section 4.2.2.

Given a supplier site $s \in S$ and a carrier $c \in C$, let $\mathrm{ST}(s, c)$ denote the shipment time distribution associated with this site–carrier pair. Let $t = t^s - t_1 \in T_0^s \subset \mathbb{R}_+$ represent the elapsed time since the shipment was initiated to the estimation time. Using these definitions, we formalize the Transit Time (TT) as follows:

- **Pointwise** TT:

$$\mathrm{TT}_\mu(s, c, t) = \max\left(\mathrm{ST}_\mu(s, c) - t,\ 0\right), \quad \mathrm{TT}_\mu : S \times C \times \mathbb{R}_+ \to \mathbb{R}_+$$

  where $\mathrm{ST}_\mu(s, c)$ is the expected shipment time for the given (s, c) pair.

- **Interval-based** TT:

$$\mathrm{TT}_\beta(s, c, t) = \max\left(\mathrm{ST}_\beta(s, c) - (t, t),\ 0\right)$$
$$\mathrm{TT}_\beta : S \times C \times \mathbb{R}_+ \to \mathbb{R}_+^2$$

  where $\mathrm{ST}_\beta(s, c) = \left(L_\beta^{ship}(s, c),\, U_\beta^{ship}(s, c)\right)$ is the β-level confidence interval of shipment time, and the maximum is applied component-wise to ensure non-negativity.

This indicator depends only on the *(site, carrier)* pair and the elapsed shipment time t, proving an estimation of the remaining shipment duration independent of the shipment network topology or the current location v.

Path Time

Consider a *maximal path* $\pi = (v_0, v_1, \ldots, v_k = m)$, along with the *shipment event time* t_e^s and the *shipment estimation time* t^s.

Our goal is to define a function that estimates the total time required to traverse the shipment path π, starting from the current vertex $v = v_0$ at time t^s and proceeding to the manufacturer m.

To this end, we introduce the Path Time (PT) function as a recursive formulation that accumulates both travel and residence times along the path. At each step, the current time t is updated to reflect the delays incurred, allowing the estimate to incorporate the temporal dynamics of the shipment process.

$$\mathrm{PT}_\mu : V^{k+1} \times \mathbb{R}_+ \to \mathbb{R}_+$$

$$
\begin{aligned}
&\mathrm{PT}_\mu(\pi, t^s) = \mathrm{PT}^v_\mu(\pi, 0, t^s) \\
&\mathrm{PT}^v_\mu(\pi, i, t) = \begin{cases} 0, & \text{if } v_i = m, \\ vt + \mathrm{PT}^r_\mu(\pi, i, t + vt), & \text{otherwise} \end{cases} \\
&\mathrm{PT}^r_\mu(\pi, i, t) = rt + \mathrm{PT}^v_\mu(\pi, i+1, t + rt)
\end{aligned}
$$

where:

$$
\begin{aligned}
vt &= \begin{cases} \max\left(\mathrm{VT}_\mu(v_i) - (t^s - t^s_e), 0\right), & \text{if } i = 0, \\ \mathrm{VT}_\mu(v_i), & \text{otherwise} \end{cases} \\
rt &= \mathrm{DRT}_\mu(v_i, v_{i+1}, t)
\end{aligned}
$$

The correction involving t^s_e in the first vertex time accounts for the amount of time already elapsed $t^s - t^s_e$ at the initial vertex.

In other words, this definition constructs a non-decreasing sequence of time estimates corresponding to each step of the shipment along the path:

$$
\begin{array}{rccccccc}
\textit{Path:} & \pi = (& v_0 = v, & v_1, & v_2, & \dots, & v_k = m &) \\
\textit{Arrival times:} & a = (& a_0 = t^s, & a_1, & a_2, & \dots, & a_k = t^s + \mathrm{PT}_\mu(\pi, t^s) &) \\
\textit{Departure times:} & d = (& d_0, & d_1, & d_2, & \dots, & - &)
\end{array}
$$

where

- $t_1 \le a_i \le a_{i+1} \quad \forall i = 1, \dots, k$;
- $t_1 \le d_i \le d_{i+1} \quad \forall i = 1, \dots, k$;
- $a_i \le d_i \quad \forall i = 1, \dots, k$.

However, a punctual estimation defined as above would result in unreliable results due to the variability contained inside the ORI and OTI. To overcome this limitation, we exploit the interval-based definitions of VT and DRT, defining:

$$
\mathrm{PT}_\beta : V^{k+1} \times \mathbb{R}_+ \to \mathbb{R}^2_+, \quad \mathrm{PT}_\beta = \begin{pmatrix} \mathrm{PT}^u \\ \mathrm{PT}^l \end{pmatrix}
$$

$$
\begin{aligned}
&\mathrm{PT}_\beta(\pi, t^s) = \mathrm{PT}^v_\beta(\pi, 0, t^s) \\
&\mathrm{PT}^v_\beta(\pi, i, t) = \begin{cases} (0, 0), & \text{if } v_i = m, \\ vt + \mathrm{PT}^r_\beta(\pi, i, t + vt), & \text{otherwise} \end{cases} \\
&\mathrm{PT}^r_\beta(\pi, i, t) = rt + \mathrm{PT}^v_\beta(\pi, i+1, t + rt)
\end{aligned}
\tag{4.4}
$$

with:

$$
\begin{aligned}
vt &= \begin{cases} \max\left(\mathrm{VT}_\beta(v_i) - (t^s - t^s_e), 0\right), & \text{if } i = 0, \\ \mathrm{VT}_\beta(v_i), & \text{otherwise} \end{cases} \\
rt &= \mathrm{DRT}_\beta(v_i, v_{i+1}, t)
\end{aligned}
$$

and all operations are understood to be applied component-wise to scalars and vectors, for notational convenience.

After this, the sequence of punctual time estimates turns into a sequence of interval time estimates:

$$\begin{array}{rlccccc} \textit{Path:} & \pi = (& v_0 = v, & v_1, & v_2, & \ldots, & v_k = m &) \\ \textit{Arr. times:} & a = (& a_0 = t^s, & a_1, & a_2, & \ldots, & a_k = t^s + \mathrm{PT}_\beta(\pi, t^s) &) \\ \textit{Dep. times:} & d = (& d_0, & d_1, & d_2, & \ldots, & - &) \end{array}$$

where each vector is defined as

$$a_i = \begin{pmatrix} a_i^l \\ a_i^u \end{pmatrix}, \quad d_i = \begin{pmatrix} d_i^l \\ d_i^u \end{pmatrix}, \quad i = 0, \ldots, k$$

and the following additional constraints hold

- $a_i^l \leq a_i^u$;
- $d_i^l \leq d_i^u$.

Probabilistic Interpretation. It should be noted that, while the individual VT and RT indicators maintain the formal confidence level interpretations, the recursive aggregation in Equation (4.4) results in intervals that are heuristic combinations rather than strict confidence intervals. In fact, the successive addition of confidence intervals is not guaranteed to preserve the original confidence level due to potential dependencies between the values of VT and RT along the path. Therefore, the resulting PT intervals should be interpreted as *uncertainty bounds* rather than rigorous statistical confidence intervals [15].

Total Forecasted Shipment Time

Finally, we combine the PT and TT indicators to define the Total Forecasted Shipment Time (TFST) as a convex combination guided by the weighting parameter $\alpha \in [0, 1]$:

- *Pointwise* TFST:

$$\begin{aligned} &\mathrm{TFST}_\mu : \mathbb{R}_+ \times \mathbb{R}_+ \times [0,1] \to \mathbb{R}_+, \\ &\mathrm{TFST}_\mu = \alpha \cdot \mathrm{TT}_\mu + (1-\alpha) \cdot \mathrm{PT}_\mu \end{aligned} \tag{4.5}$$

- *Interval-based* TFST:

$$\begin{aligned} &\mathrm{TFST}_\beta : \mathbb{R}_+^2 \times \mathbb{R}_+^2 \times [0,1] \to \mathbb{R}_+^2, \\ &\mathrm{TFST}_\beta = \alpha \cdot \mathrm{TT}_{\beta_1} + (1-\alpha) \cdot \mathrm{PT}_{\beta_2} \end{aligned} \tag{4.6}$$

  Where β_1, β_2 represent respectively the TT and PT confidence intervals. As for PT, in this formulation the formal confidence level interpretation of TT is also lost, meaning that TFST should be regarded purely as a vector in $\mathbb{R}_+^2$ representing an uncertainty bound, with no strict probabilistic interpretations.

The weighting factor α in the convex combination controls the relative contribution of the two indicators. The intended behavior is as follows:

- When the order is progressing on schedule (i.e., well captured by historical indicator distributions), we aim to emphasize TT by assigning it greater weight.

- When the order is ahead of schedule or delayed (i.e., poorly represented by historical distributions), we rely more heavily on dynamic characteristics of PT.

In the following, we propose two possible strategies for determining α.

Weighting Factor α

Constant. A straightforward approach is to set α to a constant value in the interval $[0, 1]$, independent of the shipment state. In particular, the most conservative choice is $\alpha = 0.5$, giving equal weight to the TT and PT indicators. However, this formulation does not exploit the strengths of each indicator and cannot adapt to varying conditions along the shipment.

Functional. Alternatively, α can be modeled as a function of the elapsed shipment time $t = t^s - t_1$, based on the following observations:

- The TT cannot dynamically adapt to delays or early arrivals, as it relies exclusively on historical data rather than the updated status of the shipment.
- The PT incorporates any delays accumulated during the shipment each time a new tracking event is retrieved. Its variability is initially high, due to the large number and lengths of possible paths, but decreases as the shipment progresses, since both the number and the lengths of the remaining feasible paths are reduced.

Consequently, it is desirable to assign higher weight to TT at the beginning of the shipment (when limited or no delay is expected) thus reducing the uncertainty introduced by PT. As time progresses, α should decrease to increase reliance on PT, which better reflects real-time deviations.

To incorporate these observations, we can define α as a non-increasing function of the elapsed shipment time:

$$\alpha = f(t), \quad f : T_0^s \to [0, 1]$$

respecting the monotonicity condition

$$\alpha(t) \geq \alpha(t + \Delta t), \quad \forall t, \Delta t \geq 0$$

which ensures that the influence of PT increases as the shipment progresses, while the weight of TT correspondingly decreases.

4.3.5 Graph Level Indicators

Expected Path Time

Let

- $\boldsymbol{\pi} = (\pi_1, \pi_2, \ldots, \pi_\ell)$ be all the paths extracted from a given vertex v to the manufacturer vertex m;
- $\boldsymbol{p}^c = (p_1^c, p_2^c, \ldots, p_\ell^c) \in [0, 1]^\ell, \sum_{i=1}^{\ell} p_i^c = 1$ be the corresponding probability distribution given a specific carrier c;

- X be a *random path* with $P(X = \pi_i) = p_i^c$ for $i = 1, \ldots, \ell$.

Then we define the Expected Path Time (E-PT) as the *expectation* of the PT over the set of paths $\boldsymbol{\pi}$, with respect to the probability distribution $\boldsymbol{p}^c$:

$$\underbrace{\mathbb{E}_{X\sim\boldsymbol{p}^c}\left[\mathrm{PT}(X)\right]}_{\textit{Expected Path Time}} = \sum_{i=1}^{\ell} p_i^c \cdot \mathrm{PT}(\pi_i) \tag{4.7}$$

where the operation is performed component-wise when $\mathrm{PT}(\pi_i)$ represents interval bounds:

$$\mathbb{E}[\mathrm{PT}_\beta(X)] = \left(\sum_{i=1}^{\ell} p_i^c \cdot \mathrm{PT}^l(\pi_i), \sum_{i=1}^{\ell} p_i^c \cdot \mathrm{PT}^u(\pi_i)\right)$$

This formulation captures the weighted average of path time estimates across all alternative paths, based on the likelihood that each path is followed (e.g., based on historical carrier behavior). When the interval-based PT is used, the result provides bounds that reflect both the uncertainty within individual paths and the variability across different possible routing decisions.

Expected Total Forecasted Shipment Time

We note that the TT indicator, as defined in section 4.3.4, depends only on the order supplier site s, the carrier c and the elapsed shipment time $t^s - t_1$, resulting independent of the specific path selected.
The same is valid for the weighting factor α which, as defined in section 4.3.4, depends on:

- Nothing for the constant alpha case;
- The *elapsed shipment time* $t^s - t_1$ for the functional alpha case

but never on the specific path, since all the paths start in v and end in m.

Therefore, both TT and α are constant with respect to the path distribution: as a consequence, according to the Section 4.3.5, we can exploit the linearity of the expectation operator to define the Expected Total Forecasted Shipment Time (E-TFST) as:

$$\begin{aligned}\mathbb{E}_{X\sim\boldsymbol{p}^c}\left[\mathrm{TFST}(X)\right] &= \mathbb{E}_{X\sim\boldsymbol{p}^c}\left[\alpha\cdot\mathrm{TT} + (1-\alpha)\cdot\mathrm{PT}(X)\right] \\ &= \alpha\cdot\mathrm{TT} + (1-\alpha)\cdot\mathbb{E}_{X\sim\boldsymbol{p}^c}\left[\mathrm{PT}(X)\right] \\ &= \alpha\cdot\mathrm{TT} + (1-\alpha)\cdot\sum_{i=1}^{k} p_i^c\cdot\mathrm{PT}(\pi_i)\end{aligned} \tag{4.8}$$

where, as before, the computation is carried out component-wise when $\mathrm{PT}_\beta(\pi_i)$ denotes interval bounds.

This formulation aggregates the contributions of PT and TT over the distribution of possible paths, producing a bounded expectation for the shipment time that captures the uncertainty inherent in the process.

Estimated Overall Delivery Time

Finally, we can leverage the previous defined indicators in order to estimate the value of the unknown t_n: in particular, we define the Estimated Overall Delivery Time (EODT) as the sum of EDT and the E-TFST:

- *Pointwise* EODT:

$$\mathrm{EODT}_\mu = \mathrm{EDT}_\mu + \mathbb{E}[\mathrm{TFST}_\mu]$$

  providing a direct estimation for t_n in the form:

$$t_n = t_0 + \mathrm{EODT}_\mu$$

- *Interval-based* EODT:

$$\mathrm{EODT}_\beta = \begin{pmatrix} \mathrm{EDT}_\mu \\ \mathrm{EDT}_\mu \end{pmatrix} + \mathbb{E}[\mathrm{TFST}_\beta] = \begin{pmatrix} \mathrm{EODT}^u \\ \mathrm{EODT}^l \end{pmatrix}$$

  which provides a more robust estimate by incorporating an uncertainty interval for the delivery time:

$$\begin{pmatrix} t_n^u \\ t_n^l \end{pmatrix} = \begin{pmatrix} t_0 + \mathrm{EODT}^u \\ t_0 + \mathrm{EODT}^l \end{pmatrix}$$

Chapter 5

Implementation of Indicators

This chapter presents concrete methodologies for implementing both historical and realtime indicators introduced in the previous theoretical framework. The proposed implementation addresses the practical challenges associated with limited datasets while providing a robust foundation for monitoring supply chain performance.

The chapter is organized into two main sections. The first section focuses on the implementation of historical indicators, while the second addresses realtime indicators. For the DRT implementation, machine learning techniques are employed to capture complex relationships among multiple factors influencing travel time predictions. Appendix B.2 provides an overview of the statistical foundations underlying our analytical approaches.

The implementations presented in this chapter represent a preliminary solution for operationalizing the indicator framework, providing a foundation that can be refined and extended in future stages of the project as additional data becomes available.

5.1 Implementation of Historical Indicators

In this section, we propose our solution for the concrete implementation of the historical indicators illustrated in Section 4.2.

5.1.1 Modeling Approach

Given the limited amount of data available, we decided not to search for the best-fitting probability distribution for each dispatch and shipment case individually. Such an approach would not only have required a substantially larger dataset to ensure statistical reliability of the fitting criterion but would also have increased the complexity of the model. Instead, to maintain parsimony and robustness, we opted to adopt a single probability distribution for both dispatch and shipment times, defined over the set of positive times.

Based on a review of the literature, we found several examples in [5], [8], [9] where the *Gamma distribution* has been successfully employed to model time-related processes in supply chains, which led us to choose it for our model as

well. In the following, we describe the methodology applied to construct and fit the distributions from the available data.

Fitting Procedure and Thresholds

In practice, we fitted a Gamma distribution only for samples containing at minimum of 30 observations (as a common rule of thumb in statistics), in order to ensure sufficient statistical reliability. Under this criterion, we were able to fit a single distribution for both dispatch and shipment times at supplier site 3 (located in Thief River Falls, Minnesota, US), based on 53 observations, all corresponding to shipments handled by UPS. For all other sites where the number of observations was insufficient, we relied on descriptive statistics such as the mean, variance, and empirical quantiles.

As additional data becomes available, our objective is to fit one distribution for:

- Each supplier site for dispatch times;
- Each supplier site-carrier pair for shipment times,

while continuing to use raw descriptive statistics only for newly observed sites or carriers with insufficient data.

5.1.2 Dispatch Time

Calculation Procedure

Dispatch times are obtained from the combination of pilot data (Section 3.1) and tracking data (Section 3.2). Formally, the raw dispatch time is the difference between the timestamp of the first tracking event t_1 (representing the carrier pickup) and the order creation time t_0, corresponding to the `createdAt` column of Table 3.1.

Data Cleaning

To isolate effective working days, we removed non-working days of suppliers. This was achieved by following an approach resembling the inverse of Algorithm 2, by checking for holidays within the interval $[t_0, t_1]$ and subtracting one day per holiday, following the assumptions stated in Section 3.5.

Figure 5.1 illustrates the effect of holiday cleaning on the global dispatch times distribution, with the global mean dropping from 73.92 hours to 52.44 hours.

After the initial cleaning, we grouped the data by site and performed outlier removal using the interquartile range (IQR) method for samples containing at least 10 observations. Only values within the interval

$$[q_1 - 1.5 \cdot \text{IQR},\ q_3 + 1.5 \cdot \text{IQR}], \quad \text{IQR} = q_3 - q_1$$

were retained, where q_1 and q_3 denote the first and third quartiles, respectively. Similarly as before, the threshold of 10 observations was chosen as a common rule of thumb found in the literature to ensure that the IQR-based outlier detection remains statistically meaningful for small samples.

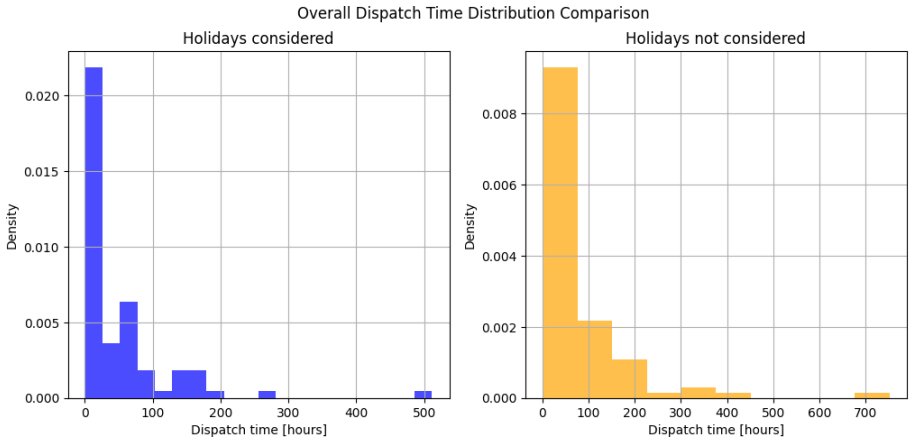


Figure 5.1: Distribution of overall dispatch times before and after accounting for holidays.

Preliminary Fit Evaluation

For eligible samples, we fitted a Gamma distribution via MLE, estimating all three parameters. Table 5.1 reports the parameters obtained with this procedure.

Parameter	Estimated Value
Shape (k)	0.8577
Scale (θ)	28.3631
Location (μ)	1.1999

Table 5.1: Gamma distribution parameters for dispatch times of site 3.

To assess the quality of the fitted Gamma distribution, we employed two complementary approaches:

- **Visualization:** We compared the fitted distribution against the empirical data using Quantile-Quantile (Q-Q) plots and overlaid histograms, providing a visual check of the goodness of fit.
- **Bootstrap Kolmogorov-Smirnov Test:** To account for the uncertainty introduced by estimating parameters from the data, we applied a bootstrap version of the K-S test. The procedure generates 1000 synthetic datasets from the fitted Gamma distribution, re-estimates parameters for each, and computes the corresponding KS statistics. The p-value is the proportion of bootstrap statistics exceeding the observed KS statistic. Formally, the hypotheses remain:

  H_0 : The sample follows a Gamma distribution,

  H_1 : The sample does not follow a Gamma distribution.

  A high p-value indicates insufficient evidence to reject H_0, while a low p-value suggests the Gamma model may not be appropriate.

The corresponding plots in Figures 5.2 and 5.3 indicate a reasonable, though not perfect, fit, which could likely improve with additional observations. Consistently, the bootstrap K-S test yielded a test statistic of 0.1127 and a p-value of 0.58. Since this p-value is well above common significance levels (0.01, 0.05, or 0.1), we do not reject H_0, indicating that there is no statistical evidence against the assumption that the dispatch times follow a Gamma distribution. This result supports the decision to model dispatch times using a Gamma distribution.

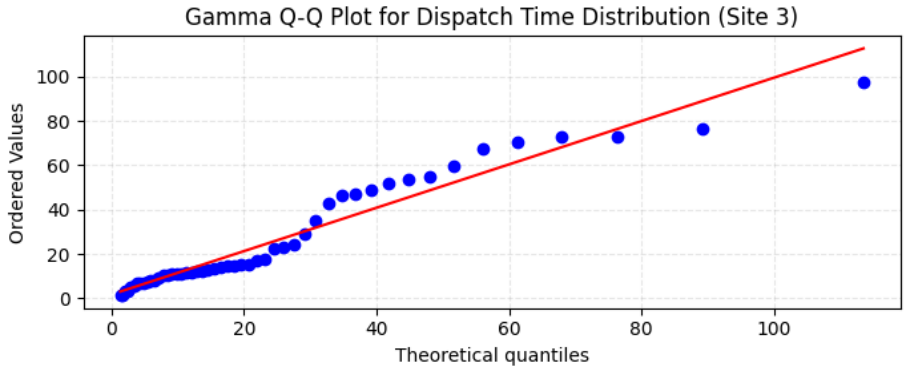


Figure 5.2: Q-Q plot of dispatch times for site 3 compared to the fitted Gamma distribution.

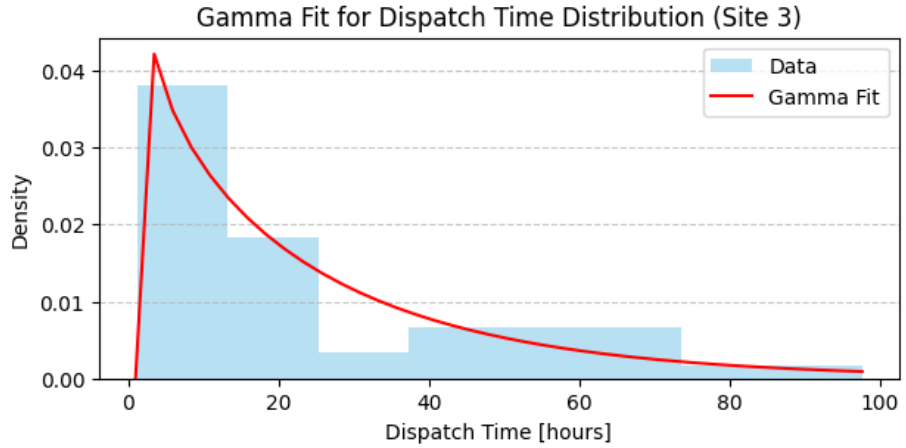


Figure 5.3: Histogram of dispatch times for site 3 compared to the fitted Gamma distribution.

5.1.3 Shipment Time

Calculation Procedure

Shipment times were derived directly from carrier tracking data (Section 3.2) as the difference between the final event t_n corresponding to order delivery and

the first recorded shipment event t_1.

Data Cleaning

Unlike dispatch times, we assume that carriers continue operations during national holidays (Section 3.5). Therefore, shipment data required only outlier removal using the same IQR-based method and minimum sample threshold as previously described.

Preliminary Fit Evaluation

For the shipment times of the only observed pair (site 3, carrier UPS), we fitted a Gamma distribution using MLE and evaluated the fit with the same procedures as for dispatch times. The estimated parameters are reported in Table 5.2.

Parameter	Estimated Value
Shape (k)	1.2119
Scale (θ)	21.2829
Location (μ)	31.0215

Table 5.2: Gamma distribution parameters for shipment times of site 3 and carrier UPS.

The Q-Q plot (Figure 5.4) and overlaid histogram (Figure 5.5) indicate a adequate fit which, as for dispatch times, could improve with a larger sample.

Consistently, the bootstrap K-S test returned a statistic of 0.1127 and a p-value of 0.555, providing no statistical evidence against the Gamma assumption and supporting, also for shipment times, the modeling through Gamma distributions.

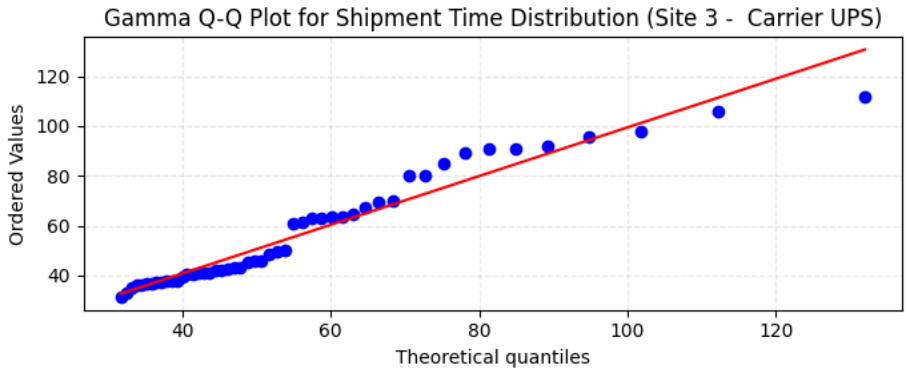


Figure 5.4: Q-Q plot of shipment times for site 3 and carrier UPS compared to the fitted Gamma distribution.

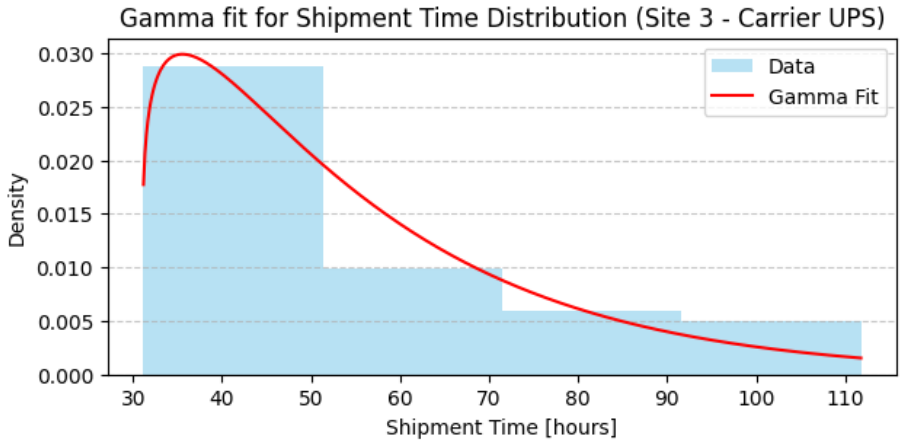


Figure 5.5: Histogram of shipment times for site 3 and carrier UPS compared to the fitted Gamma distribution.

5.2 Implementation of Realtime Indicators

Analogously to the historical indicators, this section presents the approach adopted for the concrete implementation of the real-time indicators introduced in Section 4.3.

5.2.1 Overall Residence Index

Calculation Procedure

The ORI are derived from the tracking data provided by carriers (Section 3.2). For a given vertex, an ORI observation corresponds to the total time an order spends at that location. In terms of tracking data, this is computed as the difference between consecutive timestamps recorded for the same location.

A concrete example is illustrated in Listing 2. Here, the residence time for the location *Bergamo* is obtained as the difference between the timestamp of event 2 ('2025-01-02T04:00:00Z') and event 4 ('2025-01-03T08:00:00Z'), resulting a residence time of 4 hours. If a location has only a single event (i.e., no consecutive events at the same facility), no ORI is calculated.

Data Cleaning

In contrast to dispatch times, carrier operations are assumed to remain unaffected by national holidays (see Section 3.5). Consequently, the cleaning of residence time data was restricted to outlier removal, applying the same IQR-based method and minimum sample threshold described in Section 5.1.2.

Preliminary Fit Evaluation

Applying this procedure to all available tracking data, we were able to extract samples for 36 out of 44 intermediate locations (81.82%). The remaining 8 locations never appeared twice or more consecutively in the tracking data, likely

Listing 2 Example JSON representing the steps of a shipment, with multiple updates recorded at the same location.

```
[
    {
        "timestamp": "2025-01-01T10:00:00Z",
        "location": "Gazzaniga, IT",
        "status": "Delivered"
    },
    {
        "timestamp": "2025-01-01T08:00:00Z",
        "location": "Bergamo, IT",
        "status": "InTransit"
    },
    {
        "timestamp": "2025-01-01T06:00:00Z",
        "location": "Bergamo, IT",
        "status": "Processing"
    },
    {
        "timestamp": "2025-01-01T04:00:00Z",
        "location": "Bergamo, IT",
        "status": "ArrivedAtFacility"
    },
    {
        "timestamp": "2025-01-01T00:00:00Z",
        "location": "Milan, IT",
        "status": "PickedUp"
    }
]
```

due to variations in how different carriers release tracking updates. For these intermediates without sufficient data, we assumed a negligible processing time.

Among the intermediates with data, only a small subset (4 out of 44) yielded a significant sample of 30 or more orders, corresponding to the locations:

- COLOGNE, NORTH RHINE-WESTPHALIA, DE
- TREVIOLO, LOMBARDY, IT
- SERIATE, LOMBARDY, IT
- LOUISVILLE, KENTUCKY, US

These all correspond to steps of the shipments of orders delivered from site 3, which accounts for the majority of orders in our dataset (53 out of 98). Considering the total number of observations per sample, we found:

- An average of 10.31 observations;
- A median of 2 observations

indicating that the available data is too sparse to reliably fit any probability distribution. Consequently, for this preliminary version of the model, we opted not to fit distributions for any of the vertices, instead relying on descriptive statistics such as mean, variance, and quantiles.

5.2.2 Overall Transit Index

Calculation Procedure

The OTI are computed in a manner analogous to the ORI. Specifically, the OTI for a given route represents the time taken to traverse that route. Accordingly, we derive OTI observations from consecutive events occurring at different locations, by taking the difference between their respective timestamps.

Referring again to Listing 2, it is possible to compute:

- One observation for the route $Milan \rightarrow Bergamo$, with a transit time of 4 hours;
- One observation for the route $Bergamo \rightarrow Gazzaniga$, with a transit time of 2 hours.

Data Cleaning

The same considerations outlined for ORI also apply to transit times. Accordingly, only the IQR-based outlier removal procedure described in Section 5.1.2 was applied.

Preliminary Fit Evaluation

Applying the same procedure to all available tracking data, we were able to extract at least one observation for each route. However, we encountered similar limitations as those observed for the ORI:

- Only 6 out of 82 route samples contained a sufficiently large number of observations (greater than 30), corresponding to shipments originating from the usual site 3;
- The average and median number of observations per sample (6.77 and 3, respectively) indicated that the majority of routes lacked enough data to reliably fit a distribution.

Consequently, we again opted to rely on descriptive statistical metrics rather than fitting a distribution for the OTI.

5.2.3 Traffic Meta Index

Calculation Procedure

As discussed in Section 3.4.1, the goal of the TMI is to incorporate road traffic conditions into the model in order to improve travel time estimates. While the TMI builds on traffic data retrieved from an external source, the tracking records provided no explicit indication of the transportation mode, and no

external service was available to infer it from the source, destination, and carrier. Consequently, we employed a heuristic approach to infer the transportation mode from the following set:

- Air
- Sea
- Rail
- Road (including trucks, vans, and cars)

The classification is based on constraints applied to the geodesic distance between source and destination, together with the average speed computed over that distance. Reference thresholds are reported in Table 5.3, based on values from specialized studies [34], [35], [51], [57]. We conservatively reduced some of the thresholds to account for potential delays in the publication of tracking events.

The intervals in Table 5.3 are mutually exclusive: at most one mode can be inferred for a given shipment. However, certain combinations of distance and speed may not match any category, in which case the transportation mode remains unidentified and the TMI is set to 0.

Method	**Distance [km]**	**Velocity [km/h]**
Air	$d > 500$	$v > 200$
Rail	$300 < d < 1000$	$30 < v < 100$
Road	$d < 300$	$30 < v < 110$
Sea	$d > 1000$	$10 < v < 50$

Table 5.3: Reference thresholds for transportation mode classification.

Once the transportation mode tm for a given route (v, u) is identified, the TMI is computed as:

$$\mathrm{TMI}(v,u) = \begin{cases} 0 & \text{if } tm \neq \text{road}, \\ \dfrac{r - r^*}{r} & \text{if } tm = \text{road}, \end{cases}$$

where:

- r is the estimated road travel time from v to u under *real traffic conditions*;
- r^* is the estimated road travel time from v to u under *ideal traffic conditions*.

This formulation guarantees that the computed traffic metric always lies within the interval $[0, 1]$.

Preliminary Fit Evaluation

Table 5.4 reports the results of the heuristic classification performed on our dataset. More than 90% of the shipments were assigned to either air or road transport, while only three were identified as rail, none as sea, and 43 could not

be classified given the reference thresholds. Although this distribution plausibly reflects the dominance of air and road transportation, more sophisticated approaches should be explored in the future to achieve more robust and reliable results.

Transportation Mode	**Count**	**Percentage**
Air	222	45.96%
Rail	3	0.62%
Road	215	44.51%
Sea	0	0%
Unknown	43	8.9%
Total	483	100%

Table 5.4: Distribution of inferred transportation modes.

Finally, Figure 5.6 shows the histogram of the computed TMI values for road transport. About half of the values are close to zero, while several higher values are also observed, with a peak around 0.6. These non-negligible values may capture meaningful traffic effects that can be leveraged in higher-level indicators.

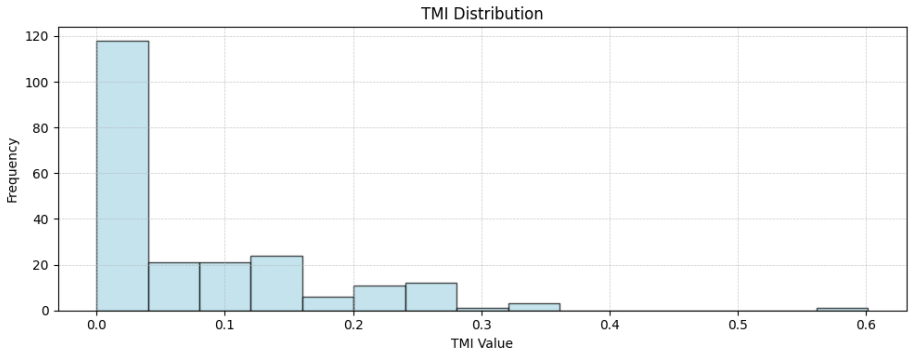


Figure 5.6: Distribution of computed TMI for road transport.

5.2.4 Weather Meta Index

Calculation Procedure

In analogy with the TMI, the supply chain model was extended to incorporate weather-related information in the estimation of travel times. Two variables were selected from the external weather source (Section 3.4.2):

- General weather conditions (e.g., sunny, cloudy, rainy, thunderstorm, etc.);
- Temperature.

Each variable was associated with a dedicated scoring function, and the overall WMI was defined as the maximum of the two:

$$\text{WMI}(x) = \max\left(wc(x),\, temp(x)\right)$$

where $wc(x)$ and $temp(x)$ denote the scores associated with weather conditions and temperature at location x, respectively.

Weather Conditions Score. Realtime weather conditions are retrieved as categorical codes (Section 3.4.2). To quantify their potential impact on shipment times, a numerical score was assigned to each code. For example:

- Clear or overcast conditions $\rightarrow$ score $= 0$;
- Rain or light snow $\rightarrow$ score $= 0.5$;
- Thunderstorms, hail, or heavy snow $\rightarrow$ score $= 1$.

The complete scoring table is available at [58].

This assignment provides an intuitive mapping of qualitative weather events to quantitative scores. However, the procedure is purely heuristic and does not rely on empirical calibration. A more robust approach would require expert validation or, ideally, the use of observed operational data to derive statistically grounded metrics.

Temperature Score. The impact of extreme temperatures on logistics performance has been emphasized in several studies (e.g., [17], [33]). To capture this effect, a piecewise logistic scoring function was introduced:

$$\text{temp\_score}(x) = \begin{cases} \dfrac{1}{1 + e^{-k_1(x - a_1)}}, & \text{if } x \geq c, \\ \dfrac{1}{1 + e^{k_2(x - a_2)}}, & \text{if } x < c \end{cases}$$

where a_1, a_2, c and $k_1, k_2 > 0$ are parameters controlling the inflection points and steepness of the curves. The functional form naturally maps values into the interval $[0, 1]$, matching the desired range.

A preliminary parameterization, chosen to yield qualitatively reasonable values without formal validation, is shown in Figure 5.7.

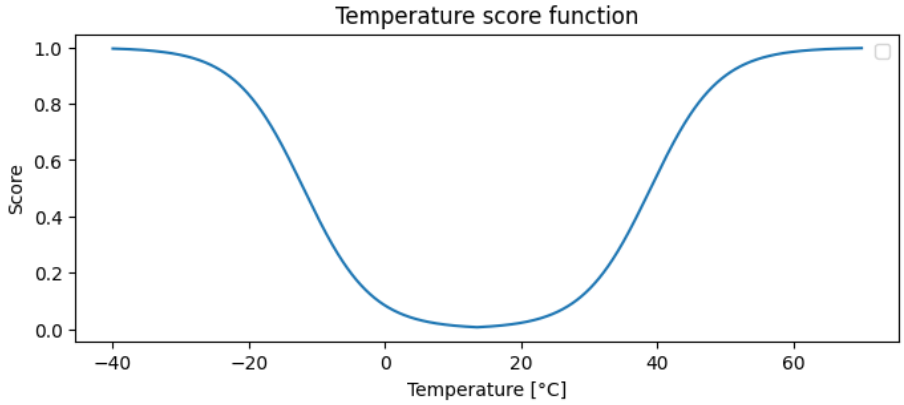


Figure 5.7: Example of a preliminary temperature score function.

This formulation is subject to several limitations, including:

- It assumes a homogeneous sensitivity to temperature across all locations. In practice, regions regularly exposed to extreme conditions are typically more resilient, and thus less affected, than regions unaccustomed to such events.
- The functional form was chosen for its boundedness in the interval $[0, 1]$, but alternative formulations could also be suitable and should be explored.
- No empirical validation has yet been performed. A proper calibration against historical data is essential to ensure robustness and reliability.

Consequently, this formulation should be regarded only as a preliminary approximation. Future work should refine it through location-specific adjustments, the consideration of alternative functional forms, and systematic data-driven validation.

Route Interpolation. Since the WMI must be evaluated over entire routes rather than single points, a procedure was developed to interpolate weather scores along the path between an origin v and a destination u. The method consists of the following steps:

1. Compute the average velocity $\overline{v}$ as the ratio between the geodesic distance d_g and the observed travel time t:
$$\overline{v} = \frac{d_g}{t}$$
2. Subdivide the route into m' equally spaced interpolation points, where $m' = \min(m,\, d_g/d)$, with d the target segment length and m the maximum number of points.
3. Assign a timestamp t_i to each interpolation point according to the travel time of its preceding segment:
$$t_i = t_0 + \frac{i \cdot d'}{\overline{v}}, \quad \text{with } d' = \frac{d_g}{m'}$$
4. Compute the pointwise WMI for each interpolation point and retain the maximum value:
$$\mathrm{WMI}(v, u) = \max\{\mathrm{WMI}(x_i)\}$$

To account for deviations from straight-line trajectories, weather data were first queried at the regional level, with fallback to specific coordinates in case of unavailable responses. The use of the maximum score, instead of the average, was motivated by the rationale that a single extreme event along the route may cause significant delays, while averaging would tend to underestimate such risks. A visual example of this procedure is provided in Figure 5.8.

Preliminary Fit Evaluation

For the experiments presented in this work, the following parameters were adopted:

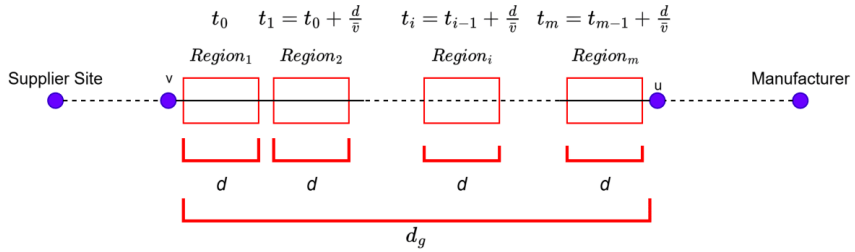


Figure 5.8: Illustration of the interpolation procedure for WMI calculation along a route.

- maximum number of interpolation points: $m = 5$;
- target sub-segment length: $d = 200\,\text{km}$.

The resulting distribution of WMI values across the dataset is shown in Figure 5.9. The histogram exhibits a marked peak at 0.5, most likely attributable to the frequent occurrence of weather conditions associated with this intermediate score. Although the approach adopted is preliminary and subject to several limitations, the resulting values may nonetheless serve as a useful basis for the construction of higher-level indicators.

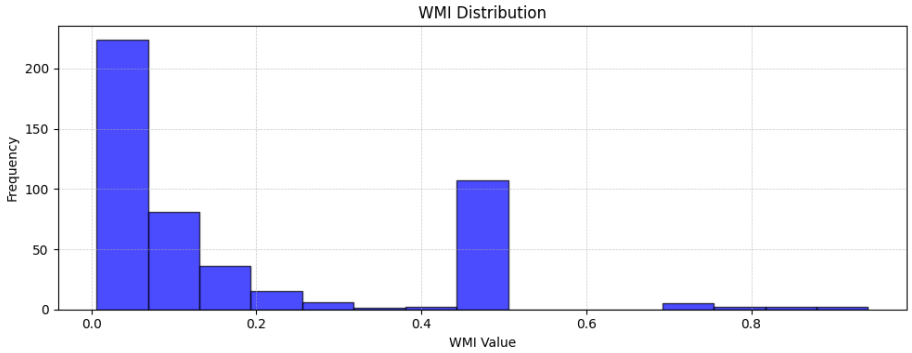


Figure 5.9: Distribution of computed WMI values over the dataset.

5.2.5 Dynamic Route Time

Calculation Procedure

We implemented the DRT as defined in Section 4.3.3 utilizing Machine Learning. In particular, we implemented the pointwise version DRT_μ by setting a regression problem where the model has to estimate the actual travel time of a given route step starting from the value of the *pointwise* RT (RT_μ), the traffic and weather scores and the some geographical information between the two points (in particular, the geodesic distance).

Later, the *Interval-based* version of the indicator (DRT_β) was implemented by constructing an uncertainty interval using a multiplicative factor γ that captures the typical deviation of predictions from observed values.

$$\mathrm{DRT}_\beta(v,u,t) = \Big[\underbrace{(1-\beta\cdot\gamma)}_{\text{lower bound}}, \underbrace{(1+\beta\cdot\gamma)}_{\text{upper bound}} \Big] \cdot \mathrm{DRT}_\mu(v,u,t)$$

where:

- $\mathrm{DRT}_\mu(v,u,t) \in \mathbb{R}_+$ is the pointwise estimate of the Dynamic Route Time, obtained from the machine learning regression model;
- $\gamma \in [0,1]$ denotes the relative margin used to define the uncertainty interval;
- the vector $\big[(1-\beta\cdot\gamma),(1+\beta\cdot\gamma)\big] \in \mathbb{R}^2_+$ defines a multiplicative uncertainty factor, forming a symmetric interval around DRT_μ scaled by $\beta \in [0,1]$.

This formulation yields an approximate uncertainty interval:

$$\mathrm{DRT}_\beta(v,u,t) = \Big[(1-\beta\cdot\gamma)\cdot\mathrm{DRT}_\mu,\ (1+\beta\cdot\gamma)\cdot\mathrm{DRT}_\mu\Big]$$

The relative margin γ was derived from the Mean Absolute Percentage Error (MAPE) observed on the test set, providing a global approximation of the model's uncertainty. Although this approach does not fully conform to the probabilistic framework described in Section 4.3.3, it offers a practical, partial solution under conditions of limited data availability. Specifically:

- The limited availability of data for each route constrains the reliability of probabilistic modeling, which affects the applicability of more sophisticated interval-prediction methods, such as quantile regression [4] or Bayesian approaches [6].
- Despite its heuristic nature, it provides a conservative uncertainty bound proportional to the observed average prediction error, allowing the model to partially account for uncertainty in travel times.
- A key limitation of this approach is that it relies on the MAPE as a global error estimate, implicitly assuming that prediction errors observed on the test set are representative across the different routes, which may not always hold in practice.
- Another limitation is the assumption of a symmetric interval around the pointwise estimate DRT_μ; in reality, prediction errors may be skewed, making the actual uncertainty potentially asymmetric.

This solution is intended as an interim measure. When more data becomes available, the previously discarded methods (quantile regression and Bayesian approaches) should be explored in order to properly fulfill the DRT definition reported in Section 4.3.3.

In the following, we present the methodology used in the DRT_μ training procedure.

Dataset and Explorative Analysis

The dataset was built leveraging the following data:

- The tracking data, providing the bases of the route times to estimate and the geographical, traffic, and weather information of the two vertices forming the route, in the form of the geodesic distance;
- The OTI distributions, used to compute the RT pointwise value (RT_μ) for each route;
- The TMI and WMI distributions, used to compute the average value of each indicator for each specific route.

As previously stated, the objective is to train a regression model DRT_μ to predict individual travel times using the features described above, which are detailed in the dataset structure in Table 5.5, comprising a limited total of 483 records.

Column	Description
d	Geodesic distance between origin and destination [km].
tmi	Traffic Meta Index computed for the route travel.
wmi	Weather Meta Index computed for the route travel.
avg_tmi	Average Traffic Meta Index over all available observations for the route.
avg_wmi	Average Weather Meta Index over all available observations for the route.
rt	Route Time computed for the route.
t	Route travel time to estimate (target variable).

Table 5.5: Description of the dataset columns.

Preprocessing

Data Cleaning. For routes with at least 10 OTI observations, an outlier detection and removal procedure based on the interquartile range (IQR) method (as described in Section 5.1.2) was applied. Extreme travel times are considered either data quality issues or exceptional events outside normal operational conditions: since the objective is to predict travel times under typical conditions, such outliers were excluded from both baseline computation and training data. This cleaning process was performed at two levels:

- **Distribution cleaning**: Outlier observations were removed from the historical OTI distributions prior to computing the route time expected values;
- **Dataset cleaning**: Records corresponding to outlier travel times were also removed from the current dataset to ensure consistency between historical baselines and prediction targets.

Routes with fewer than 10 observations were left unchanged to preserve data availability, given the already limited dataset size. Overall, this dual cleaning

procedure removed 42 records (8.7% of the dataset), leaving a total of 441 records for analysis. Figure 5.10 shows the overall distribution of the travel time target variable after the removal of outliers.

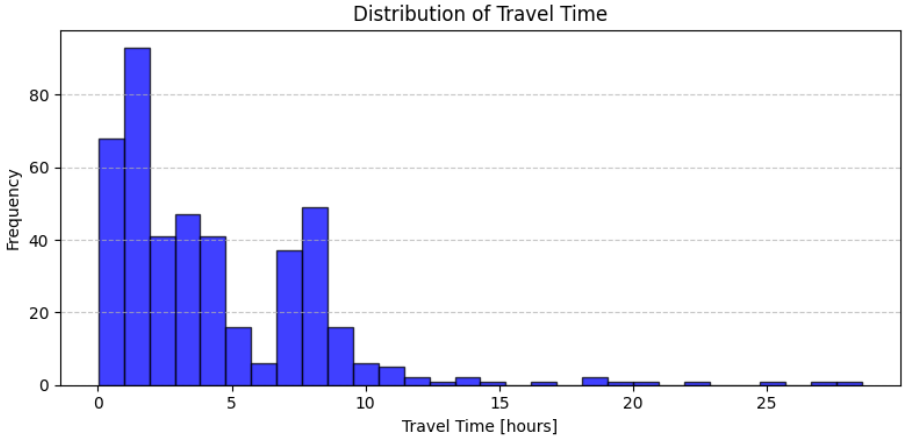


Figure 5.10: Distribution of travel times in the dataset after data cleaning.

Feature Correlation. We investigated the relationship between traffic and weather variability and travel time prediction accuracy using the Spearman rank correlation coefficient, which is suitable for non-normally distributed data and monotonic relationships. The analysis aimed to determine whether variations in traffic and weather conditions are associated with deviations of travel times from the expected RT.

First, we considered signed deviations:

- Traffic and weather deviations from route averages: $(tmi - avg\_tmi)$ and $(wmi - avg\_wmi)$;
- Travel time deviations from expected values: $(t - rt)$.

This analysis showed negligible correlations for both traffic and weather indicators, suggesting that the direction of deviation (positive or negative) is not the primary factor.

Next, we examined the magnitude of deviations using absolute values:

- Absolute deviations of traffic and weather indicators from their route averages: $|tmi - avg\_tmi|$ and $|wmi - avg\_wmi|$;
- Absolute deviation between observed and expected travel times: $|t - rt|$.

This revealed meaningful relationships: traffic deviation magnitude exhibited a moderate correlation of 0.58 with travel time deviations, whereas weather deviation magnitude showed a correlation of 0.1, indicating minimal influence. These results suggest that traffic variability significantly affects travel time predictability, independent of deviation direction, while weather variations have limited impact under the observed conditions.

Nevertheless, we retained weather data in the dataset and chose to employ machine learning models of varying complexity, as more sophisticated models might still extract useful predictive information from features with limited standalone correlation.

Dataset Splits and Data Augmentation. The dataset was partitioned into three subsets following a standard practice in supervised learning:

- **Training set**: 70% of the records (309 samples);
- **Validation set**: 10% of the records (44 samples);
- **Test set**: 20% of the records (88 samples).

The resulting test set of 88 samples is considerably small for reliable statistical evaluation of model performance. While these results can provide a preliminary indication of model behavior, conclusions should be interpreted with significant caution due to the limited sample size and potential for high variance in performance estimates.

To address the small training set size and potentially improve model generalization, we applied a *data augmentation procedure* exclusively to the training set. The method exploits route continuity: for each pair of consecutive records where the destination of the first coincides with the origin of the second, a new sample was generated by merging the two rows. The merging process was defined as follows:

- The new distance, required route time (RT), and actual travel time were computed as the sum of the corresponding values:

$$d' = d_i + d_{i+1}, \qquad rt' = rt_i + rt_{i+1}, \qquad t' = t_i + t_{i+1}$$

- For traffic and weather metrics, a weighted average was calculated, where weights correspond to the segment distances:

$$c' = \frac{c_i \cdot d_i + c_{i+1} \cdot d_{i+1}}{d'}, \quad c \in \{tmi, avg\_tmi, wmi, avg\_wmi\}$$

This approach relies on the unverified assumption that traffic and weather impacts scale linearly with segment distance, allowing for meaningful distance-weighted averaging of conditions across route segments. By applying this augmentation procedure iteratively, we obtained three versions of the dataset:

- ds_0: The original dataset, without any augmentation;
- ds_1: Resulting from the first augmentation iteration, where new samples were created by merging pairs of consecutive routes;
- ds_2: Resulting from the second augmentation iteration, where new samples were created by merging sequences of three consecutive routes.

Table 5.6 summarizes the size and composition of each dataset version.

Dataset	Size	Augmentation Iterations	Generated Samples
ds_0	441	0	0
ds_1	765	1	324
ds_2	991	2	226

Table 5.6: Summary of dataset versions after augmentation.

All models were trained and evaluated on these three dataset variants to assess the impact of data augmentation on predictive performance, acknowledging that the small test set size limits the reliability of these comparisons.

Feature Engineering. For each dataset variant, we evaluated different feature subsets to identify combinations that could enhance model performance. To avoid the exponential number of combinations that would result from testing all possible feature subsets, we selected four representative configurations, labeled A through D and reported in Table 5.7. These subsets differ in the inclusion or exclusion of geographical information and average traffic and weather indicators, resulting in a total of 12 distinct training sets.

Subset	*d*	*tmi*	*avg_tmi*	*wmi*	*avg_wmi*	*rt*	Description
A	✓	✓	✓	✓	✓	✓	All features included.
B	✓	✓	✗	✓	✗	✓	Average traffic and weather data excluded.
C	✗	✓	✓	✓	✓	✓	Geographical data excluded.
D	✗	✓	✗	✓	✗	✓	Geographical and average weather and traffic data excluded.

Table 5.7: Feature availability across dataset subsets.

These subsets were designed to test specific hypotheses about feature importance:

- **Subset A vs B**: Tests whether historical averages (avg_tmi, avg_wmi) provide additional predictive power beyond current conditions;
- **Subset A vs C**: Evaluates the contribution of geographical information (distance) to prediction accuracy;
- **Subset D**: Represents a minimal feature set to assess baseline predictive capability using only current traffic/weather conditions and historical route times.

Models and Evaluation

To enable a meaningful comparison, we first established a *baseline model* that always predicts the RT value for any given input, disregarding all other features. This naive approach serves as a reference point for assessing the added value of the proposed models.

Subsequently, as previously noted, we selected a diverse set of machine learning models with varying levels of complexity, ranging from linear to ensembling methods, in order to assess whether additional predictive value could be extracted from features with seemingly weak correlation, such as weather data. Specifically, we considered the following four widely adopted algorithms for regression tasks:

- **Linear Regression** [3];
- **Random Forest Regressor** [10];
- **Extreme Gradient Boosting (XGBoost)** [19];
- **Light Gradient Boosting Machine** (**LightGBM**) [21].

Each model was trained on all dataset variants without hyperparameter tuning at this stage; optimization was reserved only for the best-performing models identified after this phase. In total, this process resulted in 48 trained models. For XGBoost and LightGBM, early stopping based on validation performance was applied to determine the optimal number of boosting iterations. The complete training procedure, including the baseline hyperparameter configurations, is available at [49].

Model performance was evaluated using the following metrics:

- **Mean Absolute Error (MAE):**

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

- **Root Mean Squared Error (RMSE):**

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

- **Mean Absolute Percentage Error (MAPE):**

$$\text{MAPE} = \frac{100}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right|$$

Rather than prioritizing a single metric, model selection was based on consistent performance across all three measures. This approach mitigates the risk of over-optimizing for a specific error type and ensures a more balanced evaluation of accuracy and robustness.

Results

The preliminary training results are reported in Table 5.8. Several models consistently outperform the baseline across all feature subsets, supporting both the relevance of the selected features and the suitability of machine learning approaches for this task. Moreover, data augmentation yielded notable improvements across all metrics for most models, with the exception of Linear Regression, which appears to lack the complexity required to capture the underlying patterns effectively. Specifically, the first augmentation iteration provided performance gains for nearly all models, whereas the second iteration delivered benefits only for a limited subset of models while, in some cases, resulting in performance degradation.

Model	Dataset	**Subset A**			**Subset B**			**Subset C**			Subset D		
		MAE	RMSE	MAPE (%)	MAE	RMSE	MAPE (%)	MAE	RMSE	MAPE (%)	MAE	RMSE	MAPE (%)
Baseline	All	1.1776	2.0302	0.9643	1.1776	2.0302	0.9643	1.1776	2.0302	0.9643	1.1776	2.0302	0.9643
Linear Regression	ds_0	1.4014	2.0558	1.3895	1.1884	1.9532	0.6812	1.4665	2.0478	1.5430	1.2623	1.9361	0.8049
Random Forest	ds_0	1.0748	2.0248	0.7869	1.0837	2.1047	0.8400	1.0838	2.1048	0.8400	1.0942	1.8193	1.5767
XGBoost	ds_0	1.0451	2.1634	0.4940	1.1010	2.2753	0.5644	1.1010	2.2753	0.5644	1.1608	2.3298	0.7349
LightGBM	ds_0	1.3724	2.2733	1.4150	1.4249	2.3456	2.0049	1.4249	2.3456	2.0049	1.2857	2.1548	1.5421
Linear Regression	ds_1	1.3416	1.9569	1.4827	1.1700	1.8965	0.8295	1.3720	1.9670	1.5537	1.2021	1.8980	0.9062
Random Forest	ds_1	1.0455	1.9238	0.8235	0.9846	1.7008	0.9950	0.9846	1.7008	0.9950	1.0172	1.6274	1.5525
XGBoost	ds_1	0.9230	1.7246	0.4128	0.7902*	1.6647	0.4601	0.7905	1.6449	0.4685	1.0959	2.0909	1.0819
LightGBM	ds_1	1.0420	1.8618	0.7314	1.0362	1.7081	1.1943	1.0343	1.7011	1.0832	1.0405	1.6091	0.8216
Linear Regression	ds_2	1.2794	1.9078	1.5687	1.1421	1.8697	1.0047	1.3009	1.9193	1.6189	1.1642	1.8735	1.0661
Random Forest	ds_2	1.1027	2.1194	0.4577	0.9721	1.7480	0.7622	0.9721	1.7480	0.7622	1.0460	1.5908	1.6122
XGBoost	ds_2	0.9950	1.7657	0.6780	0.8464	1.5737*	0.3793*	0.8728	1.8532	0.5783	1.1352	2.1715	1.0516
LightGBM	ds_2	1.0447	1.9264	0.7371	1.0470	1.8543	1.0007	1.0470	1.8543	1.0007	1.1943	1.9036	1.0228

Table 5.8: Model performance across dataset subsets using MAE, RMSE, and MAPE (%) for all dataset versions. Best per version highlighted by color; global best marked with an asterisk (*); selected models and feature subsets shown in **bold**.

Based on the evaluation across the three metrics, XGBoost and Random Forest emerged as the best-performing models, with XGBoost providing better results in the majority of configurations across different dataset versions and feature subsets: as a consequence, we selected XGBoost for a more rigorous evaluation. Specifically, we trained two additional instances of this model on Subsets A, B, and C, excluding Subset D as it consistently underperformed across all models, suggesting that the minimal feature set lacks sufficient predictive information for this task. For each dataset, we trained two separate models:

- one using the previously established standard hyperparameters;
- one with hyperparameters automatically optimized via iterative search on the validation set.

Model evaluation was conducted using *10-fold cross-validation*, applying data augmentation exclusively to the training folds to ensure that:

- No training sample appears in the validation set;
- No augmented sample derived from validation data is included.

This guarantees full independence between training and validation folds. While this results in small validation folds, it represents the best compromise between training data availability and validation reliability under current data constraints.

For each metric, we computed *95% confidence intervals* on the Student's t-distribution, more indicated in case of small samples. Results are summarized in Table 5.9, with the first row referring to the baseline model.

Hyperparameters Optimization	Features Subset	Dataset	MAE			RMSE			MAPE (%)		
			mean	std	CI	mean	std	CI	mean	std	CI
-	-	-	1.60	0.28	[1.40,1.80]	3.58	1.50	[2.50,4.65]	0.77	0.26	[0.58,0.95]
✗	A	ds_1	1.12	0.58	[0.70,1.53]	3.01	2.84	[0.98,5.05]	0.44	0.17	[0.32,0.57]
✓	A	ds_1	1.37	0.57	[0.96,1.78]	3.15	2.79	[1.16,5.15]	2.01	0.97	[1.32,2.7]
✗	B	ds_2	1.14	0.58	[0.72,1.55]	3.02	2.63	[1.14,4.89]	0.54	0.28	[0.34,0.75]
✓	B	ds_2	1.18	0.57	[0.77,1.59]	2.93	2.31	[1.28,4.58]	0.58	0.33	[0.35,0.82]
✗	C	ds_1	1.09	0.57	[0.69,1.50]	2.84	2.74	[0.88,4.80]	0.47	0.22	[0.31,0.63]
✓	C	ds_1	1.10	0.56	[0.70,1.50]	2.79	2.81	[0.78,4.80]	0.54	0.21	[0.39,0.69]

Table 5.9: Summary statistics from 10-fold cross-validation of XGBoost with mean, standard deviation and 95% confidence interval. The first row corresponds to the baseline model; best-performing values (based on the mean) are highlighted with colors.

We first observe a noticeable deterioration in the average metric values after cross-validation compared to those computed on the test set, indicating that the test set is too small to provide reliable performance estimates. This conclusion is further supported by the wide confidence intervals and by the observation that hyperparameter optimization often reduces performance, highlighting the limited size and statistical reliability of the validation set. Acknowledging these limitations, for this preliminary version of the prototype we ultimately selected the model trained on subset A (including all features) without hyperparameter optimization. This choice was motivated by several practical considerations:

- Subset A represents the most comprehensive feature set, ensuring that the implemented system can accommodate all available data sources;
- This approach minimizes future system modifications when larger datasets become available, as the pipeline already handles the full feature space rather than requiring architectural changes to incorporate additional features;
- Without hyperparameter optimization, we avoid potential overfitting on the limited validation set;
- The complete feature set allows for more flexible, statistically grounded feature selection decisions as additional data, enabling more reliable analysis, becomes available.

5.2.6 TFST Weighting Factor

We now present two possible functional formulations for the weighting factor α used in the computation of the TFST, as introduced in Section 4.3.4.

Both approaches rely on the variable

$$\tau(s, c, t_1, t^s) = \frac{t^s - t_1}{\mathrm{ST}_\mu(s, c)}, \quad \tau \geq 0$$

which represents the ratio between the elapsed shipment time $(t^s - t_1)$ and the expected shipment time $\mathrm{ST}_\mu(s, c)$ for the specific site–carrier pair. Intuitively, τ expresses the fraction of expected time that has already elapsed for a shipment. This variable naturally aligns with the objective discussed in Section 4.3.4: as a shipment approaches its expected completion time (or exceeds it), the weight of PT should increase relative to TT.

In the following, we detail two candidate functional forms for α based on τ.

Linear Formulation

As a first baseline solution, we consider α as a simple linear function of τ. To ensure that α remains within the interval $[0, 1]$, the function is clipped at 0 whenever $\tau \geq 1$ (i.e., when the shipment exceeds its expected duration). The resulting formulation is:

$$\alpha(\tau) = \max(1 - \tau, 0)$$

This function satisfies the required constraints and provides an intuitive interpretation: the weight assigned to TT decreases linearly as the shipment progresses, reaching zero when the elapsed time equals or surpasses the expected time. Consequently, in the event of delays, the calculation of TFST relies entirely on PT, as intended.

Figure 5.11 illustrates the behavior of the linear weighting function across different values of τ.

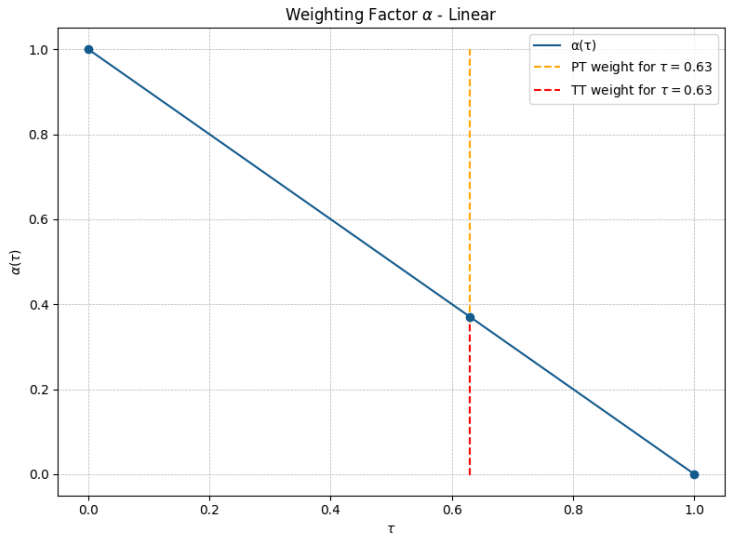


Figure 5.11: Linear functional form of the weighting factor α as a function of τ.

Polynomial Formulation

As a second approach, we explored a more flexible family of functions for the weighting factor α, consisting of polynomial forms parameterized by a factor controlling the relative weight assigned to TT over a shipment of expected duration ST_μ. Specifically, let $\langle \mathrm{TT} \rangle \in [0, 1]$ denote the target relative weight of the TT indicator. We define the following continuous family:

$$\alpha(\tau; \langle \mathrm{TT} \rangle) = \begin{cases} (1 - \tau)^q, & 0 \leq \tau < 1, \\ 0, & \tau \geq 1, \end{cases} \qquad q = \frac{1}{\langle \mathrm{TT} \rangle} - 1$$

The resulting functions are illustrated in Figure 5.12. By Theorem A.4, $\langle \mathrm{TT} \rangle$ corresponds exactly to the mean value of α in the TFST formulation, and thus to the total weight assigned to TT for a shipment of expected duration ST_μ.

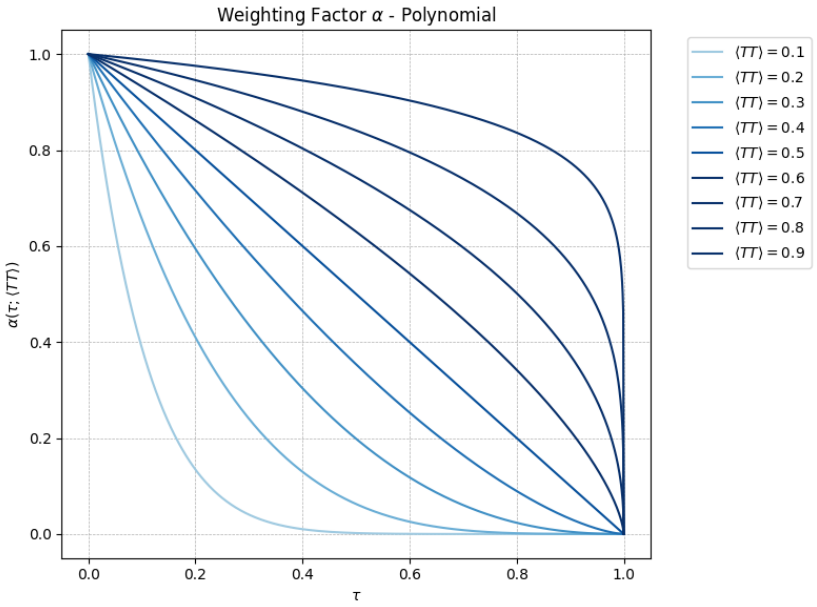


Figure 5.12: Polynomial weighting functions $\alpha(\tau; \langle \mathrm{TT} \rangle)$ for different values of $\langle \mathrm{TT} \rangle$.

For any $\langle \mathrm{TT} \rangle \in [0, 1]$, the functions satisfy the following properties:

$$\begin{aligned} &\alpha(0; \langle \mathrm{TT} \rangle) = 1 \\ &\alpha(\tau; \langle \mathrm{TT} \rangle) = 0 \quad \text{for } \tau \geq 1 \\ &\alpha(\tau; \langle \mathrm{TT} \rangle) \geq \alpha(\tau + \Delta\tau; \langle \mathrm{TT} \rangle), \quad \Delta\tau \geq 0 \end{aligned}$$

corresponding to

- Full weight to TT at the start;
- Full weight to PT at the expected end and for delays;
- Monotone non-increasing in τ.

Additionally, the limit cases highlight the flexibility of this formulation:

- $\langle \mathrm{TT} \rangle = 0.5$ recovers the linear formulation:

$$\alpha(\tau; 0.5) = (1 - \tau)^{\frac{1}{0.5} - 1} = 1 - \tau.$$

- As $\langle \mathrm{TT} \rangle \to 0$, the function becomes highly convex, concentrating the TT weight near $\tau \approx 0$ and assigning nearly full weight to PT for most of the shipment. For the limiting case $\langle \mathrm{TT} \rangle = 0$, we can simplify the function to the constant $\alpha(\tau; 0) = 0$.
- As $\langle \mathrm{TT} \rangle \to 1$, the function becomes nearly constant, assigning almost full weight to TT throughout the shipment. For the limiting case $\langle \mathrm{TT} \rangle = 1$, we can simplify it to the constant $\alpha(\tau; 1) = 1$.

Transit Time Weight Optimization. To determine an appropriate value of $\langle \mathrm{TT} \rangle$ for a given site-carrier pair, we make the following observations:

- For origins with low variability in shipment times (e.g., shipments usually arriving on time), the TT should be prioritized over PT, yielding $\langle \mathrm{TT} \rangle > 0.5$.
- Conversely, for origins with high variability in shipment times, PT should have greater influence than TT, leading to $\langle \mathrm{TT} \rangle < 0.5$.

Based on this intuition, one natural approach is to compute $\langle \mathrm{TT} \rangle$ from the variance of the shipment time distribution. For example, a clipped *coefficient of variation* in $[0, 1]$, defined as the ratio between the standard deviation and mean, could serve as a proxy for the relative weight.

However, this method does not guarantee an actual improvement in TFST estimates. Consequently, we explored an alternative approach, tuning $\langle \mathrm{TT} \rangle$ to minimize the total estimation error of the TFST relative to the observed shipment times.

Formally, given a dataset comprising:

- Actual shipment times $\{t_i = t_n^{(i)} - t_1^{(i)}\}$,
- Shipment estimation times $\{t_i^s\}$,
- Computed TT values $\{tt_i\}$,
- Computed PT values $\{pt_i\}$

for $i = 1, \dots n$, we seek the value of $\langle \mathrm{TT} \rangle$ that minimizes a loss function, such as the mean squared error

$$L(t, \hat{t}) = \frac{1}{n} \sum_{i=1}^{n} (t_i - \hat{t}_i)^2$$

where the estimated times $\hat{t}_i$ are the TFST values computed as

$$\mathrm{TFST}_\mu = \alpha(\langle \mathrm{TT} \rangle) \cdot \mathrm{TT}_\mu + (1 - \alpha(\langle \mathrm{TT} \rangle)) \cdot \mathrm{PT}_\mu$$

The optimization of $\langle TT \rangle$ constitutes a one-dimensional problem constrained to the interval $[0, 1]$, which can be efficiently solved using standard *bounded scalar optimization methods*.

Preliminary Fit Evaluation. Given the limited dataset, we performed this optimization for a single site-carrier pair (site 3, carrier UPS, with 53 observations). The result is shown in Figure 5.13, yielding an optimal value $\langle \mathrm{TT} \rangle = 0.1809$. The coefficient of variation for this sample was:

- 0.580 for the raw data,
- 0.546 for the fitted gamma distribution.

These values exceed the commonly used thresholds for classifying a high coefficient of variation [14], [50], indicating a relatively high variability in the shipment times for this origin. Although limited sample size probably influenced the high variability, this result provides an initial confirmation of the method's effectiveness, as it is consistent with the low TT weight obtained from the optimization procedure and, consequently, the higher weight assigned to the PT component.

Once more data become available, this optimization approach should be validated across multiple site-carrier pairs to assess the generalizability of the results, possibly refining the choice of the family function modeling the weighting factor.

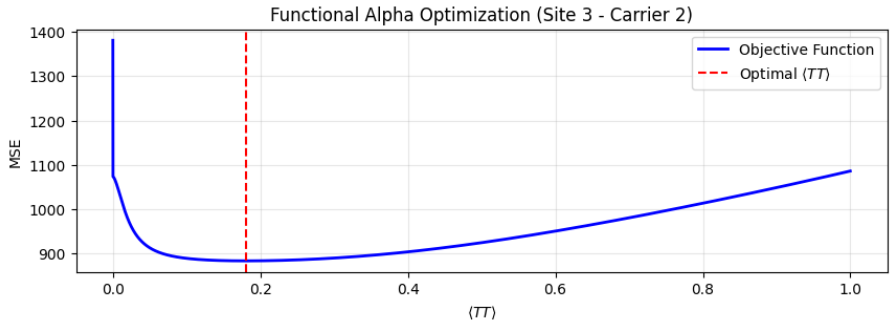


Figure 5.13: Optimized value of ⟨TT⟩ for site 3 and carrier UPS.

Chapter 6

Prototype

This chapter presents an overview of the prototype developed to implement the solutions described in the previous chapters. This module forms part of a larger system created by the broader development team: accordingly, detailed implementation aspects of the overall system fall outside the scope of this work. Instead, the focus here is on a high-level description of the components specifically designed to operationalize the proposed model, which are encapsulated within a dedicated module used to produce the results discussed in the following chapter.

6.1 Data Model

Following a company-wide decision, we adopted a relational database to implement the model and ensure seamless integration with other system components. The design of the schema focused on the following aspects:

- **Flexibility and generalizability.** The schema was designed to support future modifications with minimal effort. Although a NoSQL database could have offered greater flexibility, we opted for a relational model to ensure consistency with broader system requirements beyond the scope of this work.
- **Storage of raw data.** Instead of storing only computed indicators, we retained raw measurements (e.g., traffic and weather data) inside our database. This allows the possibility of recalculating indicators after possible future refinements of their definitions.
- **Dynamic configuration.** We introduced configuration tables to manage all model hyperparameters described in Section 6.3. This solution provides significantly greater flexibility compared to environment variables or hard-coded values, at the cost of a slight increase in schema complexity, a trade-off we considered worthwhile.

A representation of the database schema is available at [54]; some of the terminology may be unclear, as it refers to other system components not discussed in this work.

6.2 Architecture

The deployment was performed on AWS, leveraging the AWS Lambda infrastructure. This choice strongly influenced the system architecture, which was designed to integrate tightly with Lambda functions and layers. All code was implemented in Python, and resources were organized as described in the following sections. A complete overview of the prototype architecture is given in Figure 6.1, also available at [53].

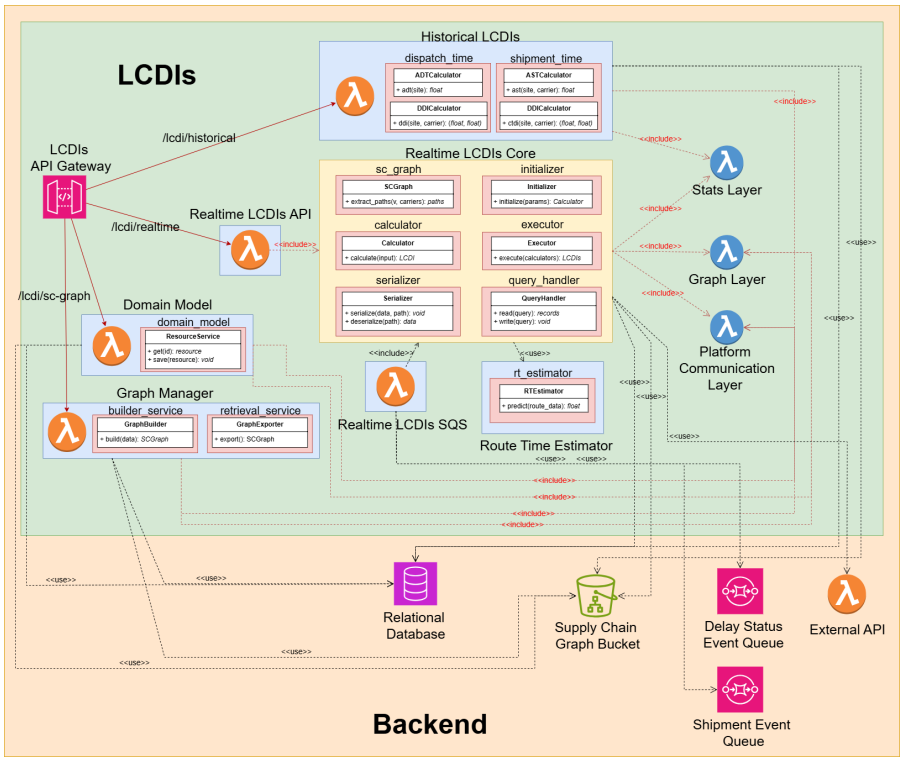


Figure 6.1: Deployment architecture of the prototype.

6.2.1 Code

The code was organized to run efficiently in the AWS Lambda environment, which abstracts infrastructure management and offers a cost-effective solution for low-traffic components. We utilized two main AWS Lambda abstractions:

- **Lambda Layers:** these were used to share common code across multiple Lambda functions. Three layers were deployed with distinct responsibilities:
  - **Stats Layer:** provides statistical functionalities, including computation of gamma distribution statistics.
  - **Graph Layer:** provides primitives for creating, modifying, and managing the SCGraph.

  - **Platform Communication Layer:** abstracts communication with databases, queues, configuration variables, and stored resources.

- **Lambda Functions:** implement the core logic of the prototype, from API request handling to indicators computation:
  - **Historical LCDIs:** manages historical indicators.
  - **Realtime LCDIs:** manages real-time indicators via two functions:
    * API access from external systems or users.
    * Internal system access using AWS Simple Queue Service (SQS) for asynchronous updates, for example when new shipment steps are retrieved.
  - **Graph Manager:** manages access and maintenance of the SCGraph.
  - **Domain Model:** provides API access to database resources.
  - **Route Time Estimator:** executes the machine learning model for estimating route times using historical, weather, and traffic data.
  - **External API:** retrieves data from external sources (part of the wider system, not strictly the prototype).

We implemented this modular design to minimize responsibilities per component, enhancing maintainability and easing the integration of future extensions.

6.2.2 API Gateway

All external API requests are routed through AWS API Gateway, which provides a unified entry point, request throttling, and security features. This allows separation between external access and internal processing, while maintaining low latency for user-facing endpoints.

6.2.3 Resource Storing

Machine learning models, serialized SCGraph instances, and other minor resources are stored in AWS S3 buckets. Lambda functions retrieve these resources on demand, ensuring that updates or retraining can be deployed without modifying the function code itself.

6.2.4 Intra-System Communication

Internal communication between Lambda functions is managed via AWS SQS queues. This design decouples components, enabling asynchronous updates–for example, triggering real-time indicator computations upon receiving new shipment events–and reduces API response latency by offloading heavy computations from synchronous request handling.

For scenarios requiring immediate responses, such as fetching traffic and weather data to compute real-time estimates, direct Lambda function invocations are employed to support synchronous communication.

6.3 Model Hyperparameters

As introduced in Section 6.1, we employed a dedicated database table to store the model hyperparameters and developed a dashboard to configure them. While a full list of possible configurations is beyond the scope of this work, the most relevant settings include:

- **Machine learning model usage for route time estimation:** Retrieving traffic and weather data is costly in terms of both computation time (due to multiple requests) and monetary cost (as external services charge beyond certain thresholds). Therefore, the ML model is used only for:
  - short-term estimations close to the shipment start time (forecasts become unreliable farther into the future);
  - the most probable extracted paths.

  In the other cases, the baseline average route time is used.
- **Weighting factor α:** The system allows dynamic selection of the implemented version of α (constant, linear, or polynomial), providing flexibility in tuning the relative contributions TT and PT.
- **Confidence thresholds:** Both historical and realtime indicator computations can be configured with confidence thresholds, allowing users to adjust sensitivity and robustness of the estimations.
- **Path extraction limits:** To limit the possible exponential growth in the number of path combinations, a maximum number of the most probable paths extracted from a source vertex to a manufacturer can be set. In this case, the path probabilities are rescaled to form a valid distribution, providing a practical heuristic solution to the potential combinatorial explosion.

Chapter 7

Experimental Evaluation

In this chapter, we present the evaluation methodology applied to assess the proposed models, together with the obtained results. Our objective is to provide an insightful and efficient evaluation of the implementations rather than relying on an exhaustive grid search across all possible configurations. We first identify the most suitable implementation of the TFST weighting factor, and subsequently evaluate the improvements achieved by incorporating machine learning techniques to account for traffic and weather data in route travel time estimation.

The results are then presented in three complementary forms: overall performance metrics, a preliminary sensitivity analysis, and visualizations of the estimated time intervals for a representative sample of orders. To assess the quality of the predicted uncertainty intervals, we employ the metrics of *sharpness*, *coverage*, and *interval score*, which are described in detail in Appendix B.3.

7.1 Evaluation Procedure

7.1.1 Estimations Granularity

All the probability distributions previously described were fitted using time data at an hourly resolution. Consequently, all time-related indicators are initially produced at this level of granularity. However, this degree of detail is excessive for the intended applications of this project[1]: for this reason, we ultimately chose to express all performance metrics in days and to consider two complementary levels of granularity:

- **Hourly granularity**: metrics are first computed using the model's raw hourly predictions, and the results are subsequently reported in days;
- **Daily granularity**: model predictions are discretized in advance to integer business days, and metrics are then evaluated directly on this coarser scale.

[1]Our industrial pilot partner, FAE Technology [55], emphasized that hourly resolution predictions are not strictly necessary in practice. Instead, they are primarily interested in the expected number of business days until order delivery, even at the cost of losing hour-level detail, provided this leads to more stable and reliable estimates.

By construction, daily-granularity intervals tend to be **wider** than those at the hourly level. This relaxation of precision is expected to yield **higher empirical coverage**, that is, more reliable calibration, at the cost of reduced sharpness. Thus, comparing the two granularities highlights the trade-off between **precision** (narrower, but riskier hourly intervals) and **robustness** (wider, but better calibrated daily intervals).

7.1.2 Operational Stage

As introduced in Section 4.3.1, we distinguish between two operational stages, each corresponding to a different point in the order lifecycle:

- **Dispatch stage**: the estimation made at the time the order is placed, where both the dispatch time and the shipment time must be predicted;
- **Shipment stage**: the estimations made after the order has been dispatched, where only the remaining shipment time needs to be predicted.

Concretely, we compute one initial estimation at order placement for the dispatch stage, and subsequently one estimation at each shipment tracking update, excluding the final delivery event. Since dispatch time introduces an additional and highly variable component, we expect estimations in the dispatch stage to be less precise than those in the shipment stage, where the uncertainty is restricted to the transport phase.

7.1.3 Set Splits

As discussed in Section 3.2, a small subset of orders was held out exclusively for testing purposes. Due to the limited size of this test set, we also report results on the training data. While this naturally carries the risk of overfitting and may lead to overly optimistic performance estimates, it nevertheless provides a sufficiently large sample for a more stable evaluation.

Table 7.1 summarizes the number of available estimates across the two operational stages for both the training and test sets.

Set	Stage	Samples
Train	Dispatch	88
	Shipment	915
Test	Dispatch	10
	Shipment	106

Table 7.1: Number of samples per operational stage across the train and test sets.

7.1.4 Evaluation Strategy and Model Variants

To assess the impact of different design choices in the implementation of the indicators, we begin by fixing both the PT and TT confidence parameters at an initial value of 0.9. Starting from this common ground, we explore a series of model variants according to the following procedure:

1. **Weighting factor parameterization.** We first compare alternative implementations of the TFST weighting factor, including:

   - relying solely on Transit Time (TT);
   - relying solely on Path Time (PT);
   - combining TT and PT with a constant weight of 0.5;
   - adopting the linear functional form for the weighting factor (Section 5.2.6);
   - adopting the polynomial functional form for the weighting factor (Section 5.2.6) for site–carrier pairs with a sufficiently large number of orders, while retaining the linear form for the others.

   Table 7.2 provides a summary of these initial configurations, including the corresponding model version names.

   In this phase, we employ the baseline DRT implementation, which always returns the mean route time without considering traffic and weather information. Its relative margin γ (Section 5.2.5) is fixed at 0.77, corresponding to the average MAPE value obtained through cross-validation.

   Note that, due to the definition of the different weighting factor implementations, version $v_{0.0}$, v_1, and v_2 produce identical estimates for the dispatch stage: in fact, for these implementations, the weighting factor is always evaluated to 1 within this stage, assigning full weight to TT as the shipment has not yet started. Consequently, in the following analysis, their dispatch stage results will be grouped together.

2. **Enhanced DRT modeling.** After selecting the best weighting factor parameterization, we fix this choice and evaluate the performance of the machine learning-based DRT model (Section 5.2.5) in comparison with the baseline. The analysis is restricted to the shipment stage, since the dispatch stage is unaffected by the DRT indicator.

3. **Preliminary sensitivity analysis.** We continue by investigating the robustness of the best-performing models by varying the confidence levels of TT and PT, studying the effect on coverage in order to evaluate calibration.

4. **Evolution of shipment estimates.** Finally, we present a set of visualizations illustrating how the time estimates evolve across the different stages of an order's lifecycle, using five representative orders from the test set.

Version		Weighting factor implementation	Weighting factor value
v_0	$v_{0.0}$	Constant	1.0
	$v_{0.1}$		0.0
	$v_{0.2}$		0.5
v_1	—	Linear	—
v_2	—	Polynomial	—

Table 7.2: Description of the different prototype parameterizations, showing weighting factor type and value.

7.2 Evaluation Results

7.2.1 Weighting Factor Parameterization

Tables 7.3 and 7.4 report the results obtained by varying the different weighting factor implementations, on the training and test sets respectively.

Version	Sharpness (days)		Coverage (%)		Interval Score (days)	
	Hourly	Daily	Hourly	Daily	Hourly	Daily
$v_{0.0}, v_1, v_2$	2.15	2.20	0.28	0.56	4.22	4.26
$v_{0.1}$	1.67	1.70	0.23	0.33	4.38	4.44
$v_{0.2}$	1.91	2.36	0.31	0.55	4.24	4.43

(a) Evaluation metrics for the **dispatch stage** across different time granularities.

Version	Sharpness (days)		Coverage (%)		Interval Score (days)	
	Hourly	Daily	Hourly	Daily	Hourly	Daily
$v_{0.0}$	1.84	1.87	0.66	0.96	1.97	2.03
$v_{0.1}$	0.80	0.91	0.42	0.71	1.56	1.73
$v_{0.2}$	1.32	1.38	0.55	0.84	1.65	1.82
v_1	1.38	1.50	0.63	0.88	1.67	1.83
v_2	1.03	1.18	0.52	0.84	1.45	1.63

(b) Evaluation metrics for the **shipment stage** across different time granularities.

Table 7.3: Performance comparison of model versions implementing different weighting factors on the **training set**, reporting evaluation metrics for both hourly and daily time granularities. Subtable (a) presents results for the **dispatch stage**, whereas subtable (b) presents results for the **shipment stage**.

Version	Sharpness (days)		Coverage (%)		Interval Score (days)	
	Hourly	**Daily**	**Hourly**	**Daily**	**Hourly**	**Daily**
$v_{0.0}, v_1, v_2$	2.87	3.00	0.30	0.50	7.81	7.44
$v_{0.1}$	1.62	1.10	0.10	0.40	7.41	6.88
$v_{0.2}$	2.25	2.80	0.30	0.50	7.59	7.24

(a) Evaluation metrics for the **dispatch stage** across different time granularities.

Version	Sharpness (days)		Coverage (%)		Interval Score (days)	
	Hourly	**Daily**	**Hourly**	**Daily**	**Hourly**	**Daily**
$v_{0.0}$	2.33	2.41	0.80	0.91	2.72	2.93
$v_{0.1}$	0.77	0.96	0.52	0.81	1.74	1.80
$v_{0.2}$	1.55	1.57	0.68	0.88	2.05	2.17
v_1	1.66	1.89	0.74	0.88	2.21	2.47
v_2	1.30	1.55	0.63	0.86	1.99	2.20

(b) Evaluation metrics for the **shipment stage** across different time granularities.

Table 7.4: Performance comparison of model versions implementing different weighting factors on the **test set**, reporting evaluation metrics for both hourly and daily time granularities. Subtable (a) presents results for the **dispatch stage**, whereas subtable (b) presents results for the **shipment stage**.

The main observations are as follows:

- As expected, estimates in the **dispatch stage** are considerably less precise than those in the shipment stage. Coverage is also markedly below the nominal confidence level in both hourly and daily settings, making the estimations of this stage insufficiently reliable.
- Focusing on the **shipment stage**, the hourly estimates appear rather unreliable, whereas the daily estimates achieve coverage levels only slightly below the nominal 90%, indicating a preliminary acceptable calibration on both training and test data.
- Models relying predominantly on TT tend to generate wider intervals. In fact $v_{0.0}$, the version based solely on TT, achieves the highest coverage across all splits and granularities, but at the cost of excessively wide intervals, as reflected by its high interval score. With more data, it is plausible that the estimated distributions could be better fitted, leading to narrower confidence intervals and improved scores.
- In contrast, models emphasizing PT produce narrower intervals, resulting in lower coverage but still improved average interval scores. Notably, in the test set, version $v_{0.1}$ achieves the best interval score across both granularities, while in the training set this role is taken by v_2.

Overall, we can conclude that with the current limited amount of data, the TT based version $v_{0.0}$ produces intervals that are too wide to be practically useful. For the remaining models, the interval score provides a useful indication of the best candidates, though the final choice should ultimately depend on the

manufacturer's specific trade-off preferences between sharpness and coverage. For the continuation of our evaluation, however, we select version v_2, as it generally achieves the best balance between coverage and interval width, which is also reflected in its interval score.

7.2.2 Enhanced DRT Modeling

We now introduce two additional model variants:

- $v_{3.0}$: a machine learning based DRT indicator that integrates both weather and traffic information. The implementation follows the approach described in Section 5.2.5, employing the XGBoost model detailed in Section Models and Evaluation. A relative margin of $\gamma = 0.44$ is used, corresponding to the average MAPE obtained through cross-validation.
- $v_{3.1}$: utilizing the same DRT approach as in $v_{3.0}$, but employing the same relative margin the baseline model ($\gamma = 0.77$). This allows a direct comparison to assess whether the machine learning approach provides tangible improvements.

The results for both the training and test sets are reported in Table 7.5.

Version	**Sharpness (days)**		**Coverage (%)**		**Interval Score (days)**	
	Hourly	**Daily**	**Hourly**	**Daily**	**Hourly**	**Daily**
v_2	1.03	1.18	0.52	0.84	1.45	1.63
$v_{3.0}$	0.87	0.91	0.35	0.76	1.37	1.54
$v_{3.1}$	0.99	1.11	0.50	0.82	1.41	1.59

(a) Evaluation metrics on the **training set** across different time granularities.

Version	**Sharpness (days)**		**Coverage (%)**		**Interval Score (days)**	
	Hourly	**Daily**	**Hourly**	**Daily**	**Hourly**	**Daily**
v_2	1.30	1.55	0.63	0.86	1.99	2.20
$v_{3.0}$	1.11	1.25	0.44	0.81	1.92	2.11
$v_{3.1}$	1.23	1.45	0.55	0.83	1.96	2.23

(b) Evaluation metrics on the **test set** across different time granularities.

Table 7.5: Performance comparison of model versions implementing different DRT, reporting evaluation metrics for both hourly and daily time granularities. Subtable (a) presents results for the **training set**, whereas subtable (b) presents results for the **test set**.

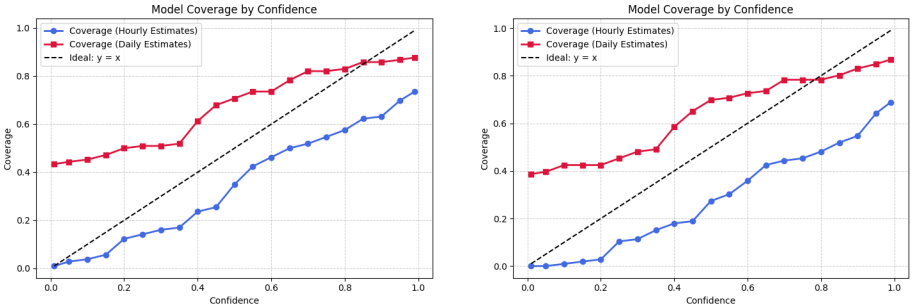
The main observations emerging are:

- Even when constrained to the same relative margin γ, the ML-based implementation consistently produces narrower prediction windows, though at the expense of reduced coverage.
- The interval score generally favors the ML-based approach, primarily due to its tighter windows. However, the practical benefit of this improvement

over the baseline remains ambiguous. Additional strategies for implementing the DRT should therefore be explored once more data becomes available.

7.2.3 Sensitivity Analysis

We continue by conducting a preliminary sensitivity analysis on models v_2 and $v_{3.1}$ with the aim of evaluating their calibration performance on the test set. Figure 7.1 reports the relationship between the nominal confidence level and the corresponding observed coverage, considering both hourly and daily estimation granularities[2].



(a) Sensitivity analysis for v_2.

(b) Sensitivity analysis for $v_{3.1}$.

Figure 7.1: Observed coverage versus nominal confidence for models v_2 and $v_{3.1}$ across both hourly and daily granularities.

Both models exhibit a very similar behavior:

- Hourly estimates fall well below the ideal trend, indicating unreliability, although the resulting curve remains roughly parallel to it.
- Daily-level estimates maintain a baseline coverage of approximately 0.4 at very low nominal confidence levels, dropping below the ideal line around a nominal confidence of 0.8, which indicates some under-reliability beyond this point.

Overall, we can conclude that the models are not reliable at the hourly granularity due to consistently low coverage, but they can provide meaningful insights when applied at the daily level, allowing for a reasonable balance between precision and calibration and making them suitable for decision-making despite some residual under-reliability at higher confidence levels.

7.2.4 Evolution of Shipment Estimates

We conclude this chapter by visualizing the evolution of time estimates produced by version $v_{3.0}$, which achieved the best interval score, for a small sample of five selected orders from the test set, ensuring that at least one order from each site is represented. Each figure is organized into two charts:

[2]Comparable studies assessing model performance across different temporal resolutions can be found in [22], [23].

- The lower chart illustrates the actual shipment time progression. The first point corresponds to the order placement, representing the dispatch stage estimate, while subsequent points correspond to the steps of the shipment process.

- The upper chart shows the deviation of the estimates from the actual delivery time. It includes the trends of the lower and upper bound estimates (labeled as CFDI in the figures, following project nomenclature) as well as the punctual estimates (EODT). Additionally, the actual delivery time and delivery day are displayed to allow interpretation at both hourly and daily granularities.

Figure 7.2 presents the shipment evolutions for orders delivered by supplier sites 4 and 1 (each with four orders), while Figure 7.3 illustrates the evolution of orders for the most-used site 3.

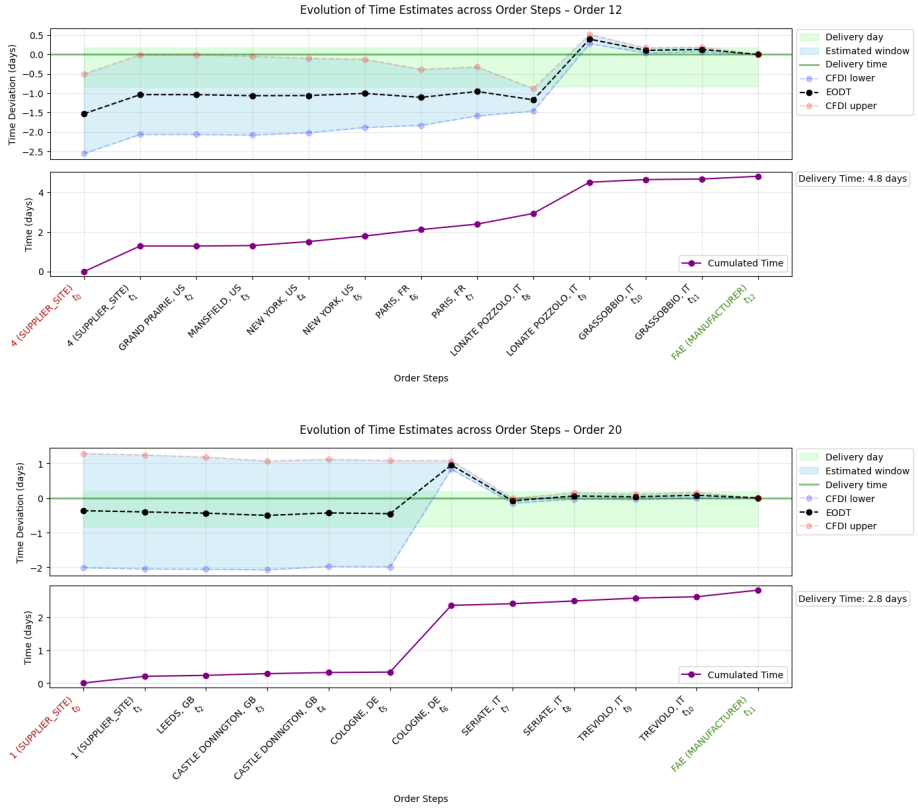


Figure 7.2: Evolution of estimations for orders 12 (site 4) and 20 (site 1).

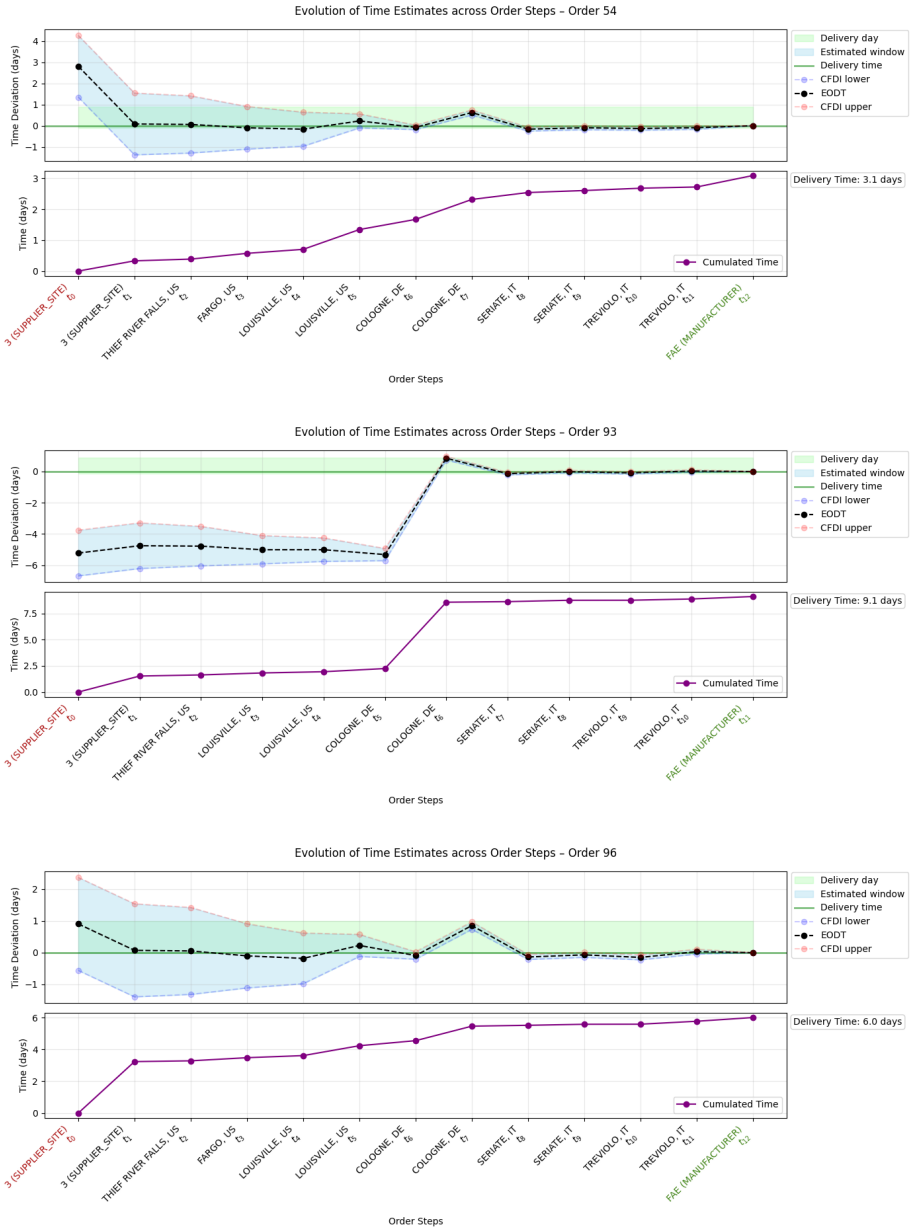


Figure 7.3: Evolution of estimations for orders 54, 93, and 96 (site 3).

Overall, the charts highlight the effectiveness of the time estimates, with most orders closely tracking the actual shipment times. Key observations include:

- **General accuracy:** The model demonstrates good performance across the visualizations, with order 93 being the only case showing significant errors during the first half steps of the shipment; however, the predictions quickly converge to the correct values.
- **Convergence of uncertainty:** In each chart, a clear convergence occurs

around the midpoint of the shipment, visible as a marked narrowing of the uncertainty interval. This behavior is likely due to the reduced number and length of paths considered by the PT, which naturally decreases uncertainty.

- **Adaptation to delays:** The model demonstrates the ability to adjust to real-time delay events, visible as steep segments in the underlying elapsed-time chart. This responsiveness represents one of its key strengths, fulfilling one of the most critical requirements of the M4ESTRO project.

Chapter 8

Conclusion

8.1 Summary of Contributions

This thesis has addressed the critical challenge of developing Logistics Chain Disruption Indicators (LCDIs) within the M4ESTRO project framework, contributing to the advancement of supply chain resilience methodologies in manufacturing environments. The work has focused specifically on delivering a comprehensive mathematical framework for representing supply chain processes and designing quantitative indicators for monitoring delivery times and detecting logistic disruptions.

The primary contributions of this work can be summarized as follows:

- **Data Integration**. The system has been designed to integrate heterogeneous data sources, combining internal operational information with external inputs such as traffic, weather, and supplier calendars. This capability ensures that disruption detection leverages a comprehensive and up-to-date view of the supply chain environment.

- **Mathematical Framework Development**. A rigorous formalization of supply chain operations has been established, incorporating the key stages of dispatch and shipment. This framework also includes the design of Logistics Chain Disruption Indicators (LCDIs), which provide quantitative measurements of supply chain performance and are intended to adapt in real time. At this stage, the focus remains on the conceptual and mathematical foundations rather than specific implementations.

- **First LCDIs Implementation**. A first implementation of the proposed indicators has been developed. Although constrained by limited data availability, this version is already operational and capable of providing valuable insights into supply chain disruptions, even if not yet at full accuracy. It serves as a solid foundation for further refinement and enhancement as richer datasets become available.

- **Prototype Development**. A functional prototype has been developed and deployed on AWS cloud infrastructure, operationalizing the theoretical framework within the M4ESTRO platform architecture. The prototype demonstrates the feasibility of real-time disruption monitoring. It

constitutes a core component of the implemented system, which in turn forms an integral part of the broader M4ESTRO ecosystem.

8.2 Key Findings

The evaluation of the developed framework has yielded several important insights regarding the practical implementation of supply chain disruption indicators:

- **Granularity Trade-offs**. The comparison between hourly and daily estimation granularities has confirmed a trade-off between precision and reliability. Hourly estimates provide finer temporal resolution but exhibit poor calibration, with coverage levels consistently below nominal confidence intervals. Daily estimates, while less precise, achieve more robust calibration and are therefore better suited for practical use. This result aligns with the industrial partners' interest in daily estimates, with no particular need for hourly precision.
- **Real-Time Disruption Detection**. The model is fully capable of detecting disruptions in real time, thereby achieving one of the most important requirements of the M4ESTRO project.
- **Operational Stage Dependencies**. The distinction between dispatch and shipment stages remains critical. Due to the scarcity of data, the model is not yet able to produce reliable results for the dispatch stage. In contrast, shipment stage estimates already provide preliminary, valuable and actionable information, benefiting from the progressive reduction of uncertainty as orders advance.
- **External Data Integration**. The current incorporation of real-time traffic and weather information through machine learning-based DRT modeling has yielded only modest improvements. However, the results indicate significant potential in the improvement of this indicator as more data become available, particularly through the exploration of more formal approaches to DRT implementation.
- **Model Calibration**. The sensitivity analysis indicates that the model's prediction intervals still suffer from under-reliability, particularly at higher confidence levels. Nevertheless, the current version already provides useful insights and a solid basis for disruption detection, while leaving room for further improvement as additional operational data is collected.

8.3 Future Developments

The work presented in this thesis opens several avenues for future research and development, both within the M4ESTRO project context and in the broader field of supply chain resilience:

- **Exploration of Improved External Data Sources**: Future work should explore and evaluate new or improved external data sources for

holidays, traffic, and weather information. Leveraging higher-quality or more comprehensive datasets could enhance the accuracy and robustness of the indicators, supporting better disruption detection and more reliable predictive performance across diverse operational scenarios.

- **Enhanced Data Integration**. As pilot integrations mature and more operational data becomes available, future work should focus on refining the probabilistic models underlying the LCDIs framework. Particular attention should be given to improving model calibration and reducing the gap between nominal and observed confidence levels.

- **Indicators Implementation Refinement**. Building on the current preliminary results with the initial implementation of the proposed indicators, future work should focus on further refining their design and performance. In particular, the two most critical indicators requiring improvement are the DRT, where advanced approaches for estimating uncertainty intervals, such as Bayesian methods and quantile regression, could enhance both reliability and interpretability, and the WMI, which should be developed further either in a fully data-driven manner or by incorporating structured expert knowledge to ensure robust integration of weather-related disruptions into the framework.

- **Multi-Manufacturer Extension**. A natural next step is to extend the framework to handle multi-manufacturer environments. In future work, the current model should be adapted to aggregate data from multiple manufacturers within a unified graph structure. This would allow the framework to collect and leverage a richer dataset while preserving the integrity and characteristics of the graph-based representation.

- **Comprehensive Validation**. As the M4ESTRO project maturates, long-term validation studies across diverse manufacturing contexts and disruption scenarios would strengthen the evidence base for the proposed approach. Such studies would be particularly valuable for understanding the framework's performance under different types of disruptions and operational conditions.

References

[1] L. R. Ford and D. R. Fulkerson, "Maximal flow through a network," *Canadian Journal of Mathematics*, vol. 8, no. 3, pp. 399–404, 1956.

[2] R. E. Tarjan, "Depth-first search and linear graph algorithms," *SIAM Journal on Computing*, vol. 1, no. 2, pp. 146–160, 1972.

[3] G. Seber and A. Lee, *Linear Regression Analysis*. Wiley, 1977.

[4] R. Koenker and G. Bassett, "Regression quantiles," *Econometrica*, vol. 46, no. 1, pp. 33–50, 1978.

[5] R. D. Snyder, "Inventory control with the gamma probability distribution," *European Journal of Operational Research*, vol. 17, no. 3, pp. 367–374, 1984. DOI: 10.1016/0377-2217(84)90133-4.

[6] R. M. Neal, *Bayesian Learning for Neural Networks*. Springer, 1996.

[7] J. R. Norris, *Markov Chains*. Cambridge University Press, 1998.

[8] K. L. Namit and J. Chen, "Solutions to the (q, r) inventory model for gamma lead-time demand," *International Journal of Physical Distribution & Logistics Management*, vol. 29, no. 6, pp. 373–387, 1999. DOI: 10.1108/09600039910264713.

[9] A. V. Hill, J. M. Hays, and E. Naveh, "A model for optimal delivery time guarantees," *Journal of service research*, vol. 2, no. 3, pp. 254–264, 2000.

[10] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[11] G. Grimmett and D. Stirzaker, *Probability and Random Processes*, 3rd. Oxford University Press, 2001.

[12] T. Gneiting and A. E. Raftery, "Strictly proper scoring rules, prediction, and estimation," *Journal of the American Statistical Association*, vol. 102, no. 477, pp. 359–378, 2007.

[13] R. Carbonneau, K. Laframboise, and R. Vahidov, "Application of machine learning techniques for supply chain demand forecasting," *European Journal of Operational Research*, vol. 184, no. 3, pp. 1140–1154, 2008.

[14] D. Faria Filho, A. Dias, A. Veloso, *et al.*, "Classification of coefficients of variation in experiments with commercial layers," *Brazilian Journal of Poultry Science*, vol. 12, pp. 255–257, 2010.

[15] R. G. Newcombe, "Propagating imprecision: Combining confidence intervals from independent sources," *Communications in Statistics-Theory and Methods*, vol. 40, no. 17, pp. 3154–3180, 2011.

[16] V. W. Berger and Y. Zhou, "Kolmogorov–smirnov test: Overview," *Wiley statsref: Statistics reference online*, 2014.

[17] J. Ludvigsen and R. Klæboe, "Extreme weather impacts on freight railways in europe," *Natural hazards*, vol. 70, no. 1, pp. 767–787, 2014.

[18] A. Seco and C. Vieira, "A multi-agent supply chain simulation analysis through a statistical mixed model," in *Procedia Technology*, Elsevier, vol. 16, 2014, pp. 163–171.

[19] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 785–794.

[20] F. Maggioni, F. Potra, and M. Bertocchi, "A scenario-based framework for supply planning under uncertainty: Stochastic programming versus robust optimization approaches," *arXiv preprint arXiv:1611.06514*, 2016.

[21] G. Ke, Q. Meng, T. Finley, *et al.*, "Lightgbm: A highly efficient gradient boosting decision tree," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017, pp. 3146–3154.

[22] Y. Huang, A. Bárdossy, and K. Zhang, "Sensitivity of hydrological models to temporal and spatial resolutions of rainfall data," *Hydrology and Earth System Sciences*, vol. 23, no. 6, pp. 2647–2663, 2019.

[23] H. Medina and D. Tian, "Comparison of probabilistic post-processing approaches for improving numerical weather prediction-based daily and weekly reference evapotranspiration forecasts," *Hydrology and Earth System Sciences*, vol. 24, no. 2, pp. 1011–1030, 2020.

[24] A. Borucka, E. Kozłowski, R. Parczewski, K. Antosz, L. Gil, and D. Pieniak, "Supply sequence modelling using hidden markov models," *Applied Sciences*, vol. 13, no. 1, p. 231, 2022.

[25] M. Liu, T. Lin, F. Chu, F. Zheng, and C. Chu, "A general robust dynamic bayesian network method for supply chain disruption risk estimation under ripple effect," *IFAC-PapersOnLine*, vol. 55, no. 10, pp. 1453–1458, 2022.

[26] A. Aljohani, "Predictive analytics and machine learning for real-time supply chain risk mitigation and agility," *Sustainability*, vol. 15, no. 20, p. 15 088, 2023.

[27] F. Bodendorf, M. Sauter, and J. Franke, "A mixed methods approach to analyse and predict supply disruptions by combining causal inference and deep learning," *International Journal of Production Economics*, vol. 256, p. 108 708, 2023.

[28] K. Sallam, M. Mohamed, and A. W. Mohamed, "Internet of things (iot) in supply chain management: Challenges, opportunities, and best practices," *Sustainable Machine Intelligence Journal*, vol. 2, pp. 3–1, 2023.

[29] A. T. Wasiac, M. S. Islamac, A. R. Akiba, and M. M. Bappybc, "Graph neural networks in supply chain analytics and optimization: Concepts, perspectives, dataset and benchmarks," *arXiv preprint arXiv:2305.12345*, 2023.

[30] X. Zhou, J. Wang, Y. Liu, X. Wu, Z. Shen, and C. Leung, "Inductive graph transformer for delivery time estimation," in *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*, 2023, pp. 679–687.

[31] S. Putthakosa and D. Hirotani, "Developing a two-stage supply chain model using a discrete-time markov chain model during supply chain disruptions," *Industrial Systems Engineering Review*, 2024, Available via ResearchGate.

[32] D. Ayrapetov, O. Larin, D. Kapski, P. Kapski, R. Khakimov, and D. Otakuziyev, "Quantifying supply chain resilience: A markov chain approach," in *ICTEA: International Conference on Thermal Engineering*, vol. 1, 2025.

[33] L. Jiang, L. Yang, X. Lyu, R. Wang, J. Guo, and Q. Wu, *Extreme heat, supply chain risks and enterprise performance.*

Sitography

[34] U.S. Department of Energy. "Fact #671: April 18, 2011, average truck speeds." (2011), [Online]. Available: https://www.energy.gov/eere/vehicles/fact-671-april-18-2011-average-truck-speeds (visited on 09/01/2025).

[35] S. Wallis. "How long does sea freight take?" World Transport Agency Group Ltd. (Aug. 2022), [Online]. Available: https://www.wtagroup.com/resources-and-insights/blogs/how-long-does-sea-freight-take (visited on 09/01/2025).

[36] D. Express. "Dhl express holiday shipping guide." (2024), [Online]. Available: https://www.dhl.com/discover/en-sg/ship-with-dhl/start-shipping/dhl-express-holiday-shipping-guide (visited on 09/01/2025).

[37] FedEx. "Holiday shipping schedule." (2024), [Online]. Available: https://www.fedex.com/en-us/holiday/last-days-to-ship.html (visited on 09/01/2025).

[38] UPS. "Ups holiday schedule." (2024), [Online]. Available: https://www.ups.com/us/en/support/shipping-support/shipping-services/holiday-shipping-schedule (visited on 09/01/2025).

[39] 17Track. "17track website." (2025), [Online]. Available: https://www.17track.net/ (visited on 09/01/2025).

[40] "17track response example." (2025), [Online]. Available: https://github.com/iFoxz17/sc-delay-prediction/blob/main/data/tracking_data_example.json (visited on 09/01/2025).

[41] A. E. AE. "Atlantis engineering ae website." (2025), [Online]. Available: https://atlantis-engineering.com/ (visited on 09/01/2025).

[42] Calendarific. "Calendarific website." (2025), [Online]. Available: https://calendarific.com/ (visited on 09/01/2025).

[43] Cefriel. "Cefriel website." (2025), [Online]. Available: https://www.cefriel.com/ (visited on 09/01/2025).

[44] C. I. Centre. "Core innovation centre website." (2025), [Online]. Available: https://www.core-innovation.com/ (visited on 09/01/2025).

[45] M. Consortium. "M4estro partners overview." (2025), [Online]. Available: https://www.m4estro-project.eu/partners/ (visited on 09/01/2025).

[46] M. Consortium. "M4estro project overview." (2025), [Online]. Available: https://www.m4estro-project.eu/ (visited on 09/01/2025).

[47] V. Crossing. "Visual crossing weather codes." (2025), [Online]. Available: https://www.visualcrossing.com/resources/documentation/weather-api/weather-condition-fields/ (visited on 09/01/2025).

[48] V. Crossing. "Visual crossing website." (2025), [Online]. Available: https://www.visualcrossing.com/ (visited on 09/01/2025).

[49] "Dynamic route time notebook." (2025), [Online]. Available: https://github.com/iFoxz17/sc-delay-prediction/blob/main/code/rt_estimator.ipynb (visited on 09/01/2025).

[50] EIKONA Logistics. "Xyz analysis in logistics." (2025), [Online]. Available: https://www.eikona-logistics.de/en/wiki/term/xyz-analysis (visited on 09/01/2025).

[51] EUROCONTROL. "Cargo and other night flights in european airspace." (2025), [Online]. Available: https://www.eurocontrol.int/sites/default/files/publication/files/tat5-night-freight-report.pdf (visited on 09/01/2025).

[52] Geonames. "Geonames website." (2025), [Online]. Available: https://www.geonames.org/ (visited on 09/01/2025).

[53] "Prototype architecture." (2025), [Online]. Available: https://github.com/iFoxz17/sc-delay-prediction/blob/main/plots/lcdi_architecture.png (visited on 09/01/2025).

[54] "Prototype data model." (2025), [Online]. Available: https://github.com/iFoxz17/sc-delay-prediction/blob/main/plots/data_model.png (visited on 09/01/2025).

[55] F. Technology. "Fae technology website." (2025), [Online]. Available: https://fae.technology/ (visited on 09/01/2025).

[56] TomTom. "Tomtom website." (2025), [Online]. Available: https://www.tomtom.com/ (visited on 09/01/2025).

[57] U.S. Department of Transportation. "Higher speed freight truck market analysis." (2025), [Online]. Available: https://rosap.ntl.bts.gov/view/dot/28268/dot_28268_DS1.pdf (visited on 09/01/2025).

[58] "Weather codes scores." (2025), [Online]. Available: https://github.com/iFoxz17/sc-delay-prediction/blob/main/data/wmi_weather_code_scores.md (visited on 09/01/2025).

Appendix A

Additional Theoretical Results

A.1 Definitions

Definition A.1 (Maximal Path)**.** Let $G = (V, A)$ be a directed graph. A path $\pi = (v_0, v_1, \ldots, v_k)$ in G is said to be *maximal* if there does not exist a vertex $v_{k+1} \in V \setminus \{v_0, \ldots, v_k\}$ such that $(v_k, v_{k+1}) \in A$. In other words, π cannot be extended by a further vertex without either forming a cycle or violating the edge constraints of the graph.

A.2 Theorems

Theorem A.1. *Let $G = (V, A)$ be a SCGraph and $v \in V$ any vertex. Then any maximal path starting from v necessarily terminates at the manufacturer node $m \in V$. Formally:*

$$\pi = (v_0, v_1, v_2, \ldots, v_k) \text{ is maximal} \iff v_k = m$$

Proof. We prove each direction of the equivalence separately.

($\Longleftarrow$) Suppose $v_k = m$. Since by construction m is a sink node with no outgoing arcs ($d_{\text{out}}(m) = 0$), the path π cannot be extended further. Hence, π is maximal.

($\Longrightarrow$) Suppose $\pi = (v_0, v_1, \ldots, v_k)$ is a maximal path and assume, for contradiction, that $v_k \neq m$.

Since m is the unique sink node in the graph, it follows that $d_{\text{out}}(v_k) > 0$. Therefore, there exists at least one outgoing arc $(v_k, v_{k+1}) \in A$.

Given that the graph is acyclic and $v_{k+1} \notin \{v_0, \ldots, v_k\}$, the path can be extended to

$$\pi' = (v_0, v_1, \ldots, v_k, v_{k+1})$$

which contradicts the assumption that π is maximal. Hence, we conclude that $v_k = m$. □

Theorem A.2. *Let $G = (V, A)$ be a SCGraph with*

$$V = S \cup I \cup M,$$
$$A = A_{S\to I} \cup A_{I\to I} \cup A_{I\to M}$$

Let $f : V^2 \to \mathbb{N}$ be a flow function satisfying the flow balance constraint at each intermediate node. Then the total flow is conserved across the graph:

$$f_V(S) = f_V(M)$$

Proof. We first compute the total outgoing flow from all supplier sites:

$$f_V(S) = \sum_{s\in S} f_V(s) = \sum_{(s,i)\in A_{S\to I}} f(s,i) \tag{A.1}$$

By the structure of the SCGraph, no arcs connect sites directly to the manufacturer, so all flow from sites must pass through intermediate nodes.

Next, the total flow from intermediate nodes to the manufacturer through $A_{I\to M}$ is:

$$f_V(M) = \sum_{(i,m)\in A_{I\to M}} f(i,m) \tag{A.2}$$

The flow balance constraint at each intermediate node $i \in I$ ensures that incoming and outgoing flows are equal:

$$\sum_{(v,i)\in A} f(v,i) = \sum_{(i,v)\in A} f(i,v), \quad \forall i \in I$$

Summing over all intermediate nodes gives:

$$\sum_{i\in I} \sum_{(v,i)\in A} f(v,i) = \sum_{i\in I} \sum_{(i,v)\in A} f(i,v)$$

We can separate flows into contributions from sites, intermediates, and manufacturer:

$$\sum_{i\in I} \sum_{(v,i)\in A} f(v,i) = \sum_{(s,i)\in A_{S\to I}} f(s,i) + \sum_{(j,i)\in A_{I\to I}} f(j,i),$$
$$\sum_{i\in I} \sum_{(i,v)\in A} f(i,v) = \sum_{(i,j)\in A_{I\to I}} f(i,j) + \sum_{(i,m)\in A_{I\to M}} f(i,m)$$

Each arc $(i,j) \in A_{I\to I}$ appears once on both sides of the equality, so these terms cancel, leaving

$$\sum_{(s,i)\in A_{S\to I}} f(s,i) = \sum_{(i,m)\in A_{I\to M}} f(i,m) \tag{A.3}$$

Combining equations (A.1), (A.2), and (A.3), we conclude that

$$f_V(S) = f_V(M)$$

demonstrating that the total flow originating from all supplier sites equals the total flow reaching the manufacturer, thus establishing flow conservation across the supply chain network. □

Theorem A.3. *Let $G = (V, A)$ be a SCGraph, let $v \in V$ be a vertex of the graph, and let $\Pi(v) = \{\pi_1, \pi_2, \ldots, \pi_\ell\}$ denote the set of all maximal paths from v to the absorbing node m. Then, the vector of probabilities $\boldsymbol{p} = (p_1, p_2, \ldots, p_\ell)$, where $p_i = \mathbb{P}(\pi_i \mid X_0 = v)$, defines a valid probability distribution:*

1. $p_i \in [0, 1] \quad \forall i = 1, \ldots, \ell$;
2. $\sum_{i=1}^{\ell} p_i = 1$.

Proof. We recall that each maximal path π_i starting from vertex v is a finite sequence

$$\pi_i = (v_0 = v, v_1, \ldots, v_{k_i} = m)$$

and its probability, conditioned on starting at v, is the product of the transition probabilities along the path:

$$p_i = \mathbb{P}(\pi_i \mid X_0 = v) = \prod_{j=0}^{k_i - 1} \mathbb{P}(X_{j+1} = v_{j+1} \mid X_j = v_j)$$

(i): Non-negativity. Since each transition probability

$$\mathbb{P}(X_{j+1} = v_{j+1} \mid X_j = v_j) = \frac{f(v_j, v_{j+1})}{f_v(v_j)} \in [0, 1]$$

it follows immediately that $p_i \in [0, 1]$ for all $i = 1, \ldots, \ell$.

(ii): Normalization. We want to prove that

$$\sum_{i=1}^{\ell} p_i = \sum_{\pi \in \Pi(v)} \mathbb{P}(\pi \mid X_0 = v) = 1$$

where $\Pi(v)$ is the set of all maximal paths starting at v. We proceed by induction on the paths first vertex v.

Base case: If $v = m$ (the absorbing node), then there is only one maximal path $\pi = (m)$, with

$$\mathbb{P}(\pi \mid X_0 = m) = P_{mm} = 1$$

so the sum over all paths starting at m is 1.

Inductive step: Consider the set of neighbors reachable in one step from v:

$$N(v) = \{u \in V : (v, u) \in A\}$$

And assume that for every vertex $u \in N(v)$ it holds

$$\sum_{\pi \in \Pi(u)} \mathbb{P}(\pi \mid X_0 = u) = 1$$

Each maximal path $\pi \in \Pi(v)$ can be uniquely decomposed as

$$\pi = (v, \pi')$$

where $\pi' \in \Pi(u)$ for some $u \in N(v)$. Using the Markov property, we can write:

$$\mathbb{P}(\pi \mid X_0 = v) = \mathbb{P}(X_1 = u \mid X_0 = v) \cdot \mathbb{P}(\pi' \mid X_0 = u) = P_{vu} \cdot \mathbb{P}(\pi' \mid X_0 = u)$$

therefore,

$$\sum_{\pi \in \Pi(v)} \mathbb{P}(\pi \mid X_0 = v) = \sum_{u \in N(v)} \sum_{\pi' \in \Pi(u)} P_{vu} \cdot \mathbb{P}(\pi' \mid X_0 = u) \tag{A.4}$$

By the inductive hypothesis, we know that

$$\sum_{\pi' \in \Pi(u)} \mathbb{P}(\pi' \mid X_0 = u) = 1$$

which allows to simplify (A.4) as

$$\sum_{\pi \in \Pi(v)} \mathbb{P}(\pi \mid X_0 = v) = \sum_{u \in N(v)} P_{vu} = 1$$

because the transition probabilities from v form a probability distribution over its neighbors. This completes the induction and proves that the path probabilities form a proper probability distribution. □

Theorem A.4. *Let $\theta \in [0, 1]$, and consider*

$$\mathrm{TFST} = \alpha(\tau; \theta) \cdot \mathrm{TT} + \big(1 - \alpha(\tau; \theta)\big) \cdot \mathrm{PT}$$

where

$$\alpha(\tau; \theta) = (1 - \tau)^q, \qquad q = \frac{1}{\theta} - 1, \qquad \tau \in [0, 1]$$

Then θ corresponds to the mean value of $\alpha(\cdot; \theta)$ over $[0, 1]$.

Proof. The *Integral Mean Value Theorem* states that for any continuous function f on $[a, b]$, there exists $\delta \in [a, b]$ such that

$$\frac{1}{b - a} \int_a^b f(x)\, dx = f(\delta)$$

Applying this to $f(\tau) = \alpha(\tau; \theta)$ on $[0, 1]$:

$$\begin{aligned}
\int_0^1 \alpha(\tau; \theta)\, d\tau &= \int_0^1 (1 - \tau)^q\, d\tau && \text{(definition of } \alpha\text{)} \\
&= \int_1^0 u^q (-du) && \text{(substitute } u = 1 - \tau,\ du = -d\tau\text{)} \\
&= \int_0^1 u^q\, du && \text{(flip the integration limits)} \\
&= \left. \frac{u^{q+1}}{q+1} \right|_0^1 && \text{(power rule)} \\
&= \frac{1}{q+1} - 0 = \frac{1}{q+1}
\end{aligned}$$

Since $q + 1 = \frac{1}{\theta}$, we obtain

$$\int_0^1 \alpha(\tau; \theta)\, d\tau = \theta$$

Hence, the mean value of α over $[0, 1]$ equals θ, corresponding exactly to the total weight assigned to TT in

$$\mathrm{TFST} = \alpha \cdot \mathrm{TT} + (1 - \alpha) \cdot \mathrm{PT}$$

□

Appendix B

Additional Material

B.1 Supporting Data Structures

For completeness, we provide an overview of the data structures employed to ensure the efficient operation of the SCGraph and the related procedures discussed in the main text. These structures include:

- A directed graph with attributes on both vertices and arcs, which stores shipment flow information as well as low-level realtime indicators. Details on these attributes and their use are provided in the main text.
- A collection of $|C|$ sparse transition probability matrices, each corresponding to the routing behavior of a specific carrier $c \in C$. The use of sparse representations is particularly advantageous due to the high sparsity of P^c_{vu}, as it allows constant-time access to nonzero entries while avoiding the memory overhead associated with dense $|V|^2$ matrices. Alternatively, transition probabilities can be computed on-the-fly from the flow information stored in the graph, reducing memory usage at the cost of additional computational effort.

In scenarios where the number of paths per vertex grows exponentially under the preceding assumptions, a possible alternative is to maintain a separate graph for each supplier. While this increases overall system complexity, it effectively bounds the number of paths considered during path extraction.

B.2 Statistical Foundations

This section provides a concise overview of the statistical tools employed in analyzing dispatch and shipment times, including the Gamma distribution, Maximum Likelihood parameter estimation, and goodness-of-fit assessment via the Kolmogorov-Smirnov test.

B.2.1 Gamma Distribution

The probability density function of the Gamma distribution with shape parameter k, scale parameter θ, and location parameter μ is given by

$$f(x; k, \theta, \mu) = \frac{1}{\Gamma(k)\,\theta^k}\,(x-\mu)^{k-1}\exp\!\left(-\frac{x-\mu}{\theta}\right), \quad x > \mu,\ k > 0,\ \theta > 0$$

where:

- $k > 0$ is the **shape parameter**, which controls skewness and tail behavior;
- $\theta > 0$ is the **scale parameter**, which stretches or compresses the distribution;
- $\mu \in \mathbb{R}$ is the **location parameter**, which shifts the distribution along the real axis;
- $\Gamma(k)$ is the **Gamma function**, defined as

  $$\Gamma(k) = \int_0^\infty t^{k-1}e^{-t}\,dt$$

  which generalizes the factorial function, i.e., $\Gamma(n) = (n-1)!$ for any integer $n > 0$.

The first two moments of the Gamma distribution are

$$\begin{aligned}\mathbb{E}[X] &= \mu + k\theta,\\ \mathrm{Var}(X) &= k\theta^2\end{aligned}$$

The probability that X takes a value less than a, i.e., the cumulative distribution function (CDF), is

$$\mathbb{P}(X < a) = F(a; k, \theta, \mu) = \frac{1}{\Gamma(k)}\int_0^{\frac{a-\mu}{\theta}} t^{k-1}e^{-t}\,dt = \frac{\gamma\big(k, \frac{a-\mu}{\theta}\big)}{\Gamma(k)}$$

where $\gamma(k, z)$ is the **lower incomplete Gamma function**. This integral generally does not have a closed form for arbitrary k, but it is implemented in most numerical libraries.

B.2.2 Maximum Likelihood Estimation

To estimate the parameters of the Gamma distribution, we applied the **Maximum Likelihood Estimation (MLE)** method. Formally, given a sample of independent observations $x_1, x_2, \ldots, x_n$, the likelihood function for the parameter vector $\boldsymbol{\theta} = (k, \theta, \mu)$ is

$$L(\boldsymbol{\theta}; x_1, \ldots, x_n) = \prod_{i=1}^{n} f(x_i; k, \theta, \mu)$$

The MLE estimator $\hat{\boldsymbol{\theta}}$ is the value that maximizes this likelihood:

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} L(\boldsymbol{\theta}; x_1, \ldots, x_n)$$

B.2.3 Kolmogorov-Smirnov Test

To assess the goodness of fit of the Gamma distribution to the observed data, we employed the **Kolmogorov-Smirnov (K-S) test** [16]. This test compares the empirical cumulative distribution function (ECDF) of the sample with the CDF of the proposed theoretical distribution. Formally, the K-S statistic is defined as the maximum absolute difference between the ECDF and the theoretical CDF:

$$D = \sup_x \left| F_n(x) - F(x; \hat{k}, \hat{\theta}, \hat{\mu}) \right|$$

where $F_n(x)$ is the ECDF of the sample and $F(x; \hat{k}, \hat{\theta}, \hat{\mu})$ is the fitted Gamma CDF.

The hypotheses of the test are:

- H_0: the sample follows the fitted Gamma distribution;
- H_1: the sample does not follow the fitted Gamma distribution.

A small value of D indicates that the sample is consistent with the theoretical distribution, whereas a large value suggests a poor fit. The corresponding p-value allows us to formally accept or reject the null hypothesis H_0.

B.3 Evaluation Metrics

To evaluate the predictive uncertainty intervals, we consider three complementary metrics: *sharpness*, *coverage*, and the *interval score*. These provide a balanced assessment of both the informativeness and the reliability of the intervals.

We'll use the following notation:

- N for the total number of predictions;
- $L = \{l_i, \ldots, l_N\}$ for the lower bounds of the predicted intervals;
- $U = \{u_i \ \ldots, u_N\}$ for the upper bounds of the predicted intervals;
- $Y = \{y_i, \ldots, y_N\}$ for the observed values;
- α for the nominal significance level of the interval (e.g., $\alpha = 0.05$ for a 95% interval);
- $\mathbf{1}\{\cdot\}$ for the indicator function, which equals 1 if the condition inside is true, and 0 otherwise.

B.3.1 Sharpness

Sharpness measures the average width of the predicted intervals, independently of the observations:

$$\text{Sharpness} = \frac{1}{N} \sum_{i=1}^{N} (u_i - l_i)$$

Narrower intervals indicate greater confidence in the estimations.

B.3.2 Coverage

Coverage quantifies the proportion of observations that fall within the predicted intervals:

$$\text{Coverage} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}\{l_i \leq y_i \leq u_i\}$$

A model whose empirical coverage is close to the nominal level $1 - \alpha$ is considered well *calibrated*, meaning that the stated confidence in its predictive intervals reliably reflects the true probability of containing the observations. Conversely, coverage significantly below or above the nominal level indicates underconfidence or overconfidence in the uncertainty estimates, respectively.

B.3.3 Interval Score

The interval score (IS) provides a single measure that accounts for both the width of the predictive interval (sharpness) and the accuracy in capturing the true observation (coverage) [12]. For each prediction i, it is defined as:

$$\text{IS}_\alpha(l_i, u_i; y_i) = (u_i - l_i) + \frac{2}{\alpha}(l_i - y_i)\mathbf{1}\{y_i < l_i\} + \frac{2}{\alpha}(y_i - u_i)\mathbf{1}\{y_i > u_i\}.$$

The first term, $u_i - l_i$, reflects the sharpness of the interval, i.e., its width. The additional terms impose a penalty when the observation y_i falls outside the interval, scaled by the significance level α. This way, narrower intervals that successfully contain the observation are rewarded, while intervals that miss the target are penalized proportionally to how far the observation lies outside the bounds.

The overall interval score is the average across all predictions:

$$\text{Interval Score} = \frac{1}{N} \sum_{i=1}^{N} \text{IS}_\alpha(l_i, u_i; y_i).$$

Lower values indicate more informative and well-calibrated predictive intervals.

B.4 Indicators Summary

Name	Scope	Type	Description
DT_{μ}	Dispatch	Scalar	Expected dispatch time for a given supplier site.
DT_{β}	Dispatch	Interval	Interval-based estimate of dispatch time at a supplier site.
ST_{μ}	Shipment	Scalar	Expected shipment time for a given (site, carrier) pair.
ST_{β}	Shipment	Interval	Interval-based estimate of shipment time for a given (site, carrier) pair.
DLT_{μ}	Delivery	Scalar	Expected total delivery time (dispatch + shipment) for a given (site, carrier) pair.
DLT_{β}	Delivery	Interval	Interval-based estimate of total delivery time for a given (site, carrier) pair.

Table B.1: Summary overview of historical indicators.

Name	Scope	Type	Description
EDT_{μ}	Vertex	Scalar	Estimated dispatch time for a given supplier site.
ORI	Vertex	Distribution	Distribution of residence times at a facility, aggregating data across carriers.
VT_{μ}	Vertex	Scalar	Expected residence time at a facility.
VT_{β}	Vertex	Interval	Interval-based estimate of the residence time at a facility.
OTI	Route	Distribution	Distribution of route times between two facilities, aggregating data across carriers.
TMI	Route	Scalar	Real-time measure of traffic congestion along a route.
WMI	Route	Scalar	Real-time measure of adverse weather impact along a route.
RT_{μ}	Route	Scalar	Expected travel time for a given route.
RT_{β}	Route	Interval	Interval-based estimate of the travel time for a given route.
DRT_{μ}	Route	Scalar	Expected travel time for a given route adjusted with real-time traffic and weather conditions.
DRT_{β}	Route	Interval	Interval-based estimate of the travel time for a given route adjusted with real-time traffic and weather conditions.
TT_{μ}	Shipment	Scalar	Expected remaining shipment time based on historical shipment data.
TT_{β}	Shipment	Interval	Interval-based estimate of remaining shipment time from historical shipment data.
PT_{μ}	Path	Scalar	Estimate of the remaining shipment time from the SCGraph model.
PT_{β}	Path	Interval	Interval-based estimate of the remaining shipment time from the SCGraph model.
TFST	Path	Scalar – Interval	Estimate of the remaining shipment time combining TT and PT.
E-PT	Graph	Scalar – Interval	Expected value of PT over all the possible paths.
E-TFST	Graph	Scalar – Interval	Expected value of TFST over all the possible paths.
EODT	Graph	Scalar – Interval	Estimated total time from order placement to delivery.

Table B.2: Summary overview of realtime indicators.