Integration and collaboration in maritime logistics and other transportation areas

Iradi, Bernardo Martin

Publication date:
2023

Document Version
Publisher's PDF, also known as Version of record

Integration and collaboration in maritime logistics and other transportation areas

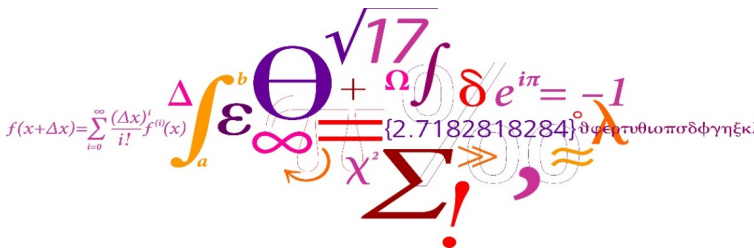
Bernardo Martín-Iradi

PhD Thesis

January 2023

DTU Management, Department of Technology, Management and Economics

Technical University of Denmark



Title:	Integration and collaboration in maritime logistics and other transportation areas
Type:	PhD Thesis
Date:	January 2023
Author:	Bernardo Martín-Iradi
Supervisors:	Dario Pacino, Associate Professor, DTU Management Stefan Røpke, Professor, DTU Management
University:	Technical University of Denmark
Department:	DTU Management, Department of Technology, Management and Economics
Division:	Division of Management Science
Address:	Akademivej Bygning 358, DTU 2800 Kgs. Lyngby, Denmark
Cover Picture:	colourbox.com

Summary

The transport sector is experiencing a rapid transformation, opening novel lines of research and creating new optimization problems to solve. This thesis aims to tackle optimization problems arising from integration and collaboration in transport and logistics. The main application area covers seaside maritime logistics, but we also consider case studies in urban transportation.

Maritime transport plays a crucial role in the global supply chain. The demand for containerized cargo is increasing, pressuring terminals and carriers to explore collaborative responses. In the first part of the thesis, we apply a collaborative approach to seaside maritime logistics, where we define a problem that jointly optimizes the scheduling of vessels and their berth allocation in terminals. The first three articles address variants of this problem and extend and improve previous studies' modeling features and performance contributions. The results contribute to quantifying the benefits of integrated and collaborative optimization and aim at supporting the transition toward collaborative and sustainable logistics.

In the second part of the thesis, we address integrated optimization in passenger transportation. The fourth article addresses on-demand microtransit, an emerging transport system that combines public transit efficiencies with the flexibility of on-demand services. While ride-sharing systems are proliferating, contributing to congestion and socio-economic inequalities, public transportation is failing to adapt to new mobility patterns. Microtransit aims to tackle these challenges by promoting efficient, sustainable, and accessible mobility. We present a unified model to design the network and optimize its operations. We demonstrate the strength of a novel modeling approach in a real-world setup, and the results validate microtransit as a sustainable and efficient mobility system. The last article focuses on railway transportation, where we study the integrated optimization of train timetabling with passenger routing. We present modeling and methodological contributions, and the results highlight the positive value of integration for passengers and operators.

Resumé (Danish)

Transportsektoren oplever en hastig forandring, hvilket åbner nye forskningsmuligheder og skaber nye optimeringsproblemer at løse. Denne afhandling har til formål at takle nye optimeringsproblemer, der opstår ved integration og samarbejde i transport og logistik. Det primære anvendelsesområde dækker maritim logistik, men afhandlingen dækker også casestudier indenfor offentlig transport.

Maritim transport spiller en vigtig rolle i den globale forsyningskæde. Efterspørgslen for containergods er stigende, hvilket presser terminaler og transportører til at udforske samarbejdsmuligheder. I den første del af afhandlingen, anvender vi en kollaborativ tilgang til maritim logistik, hvor vi definerer et problem, der optimerer tidsplanlægning af skibe og deres kajtildeling i terminaler. De første tre artikler omhandler varianter af dette problem og udvider og forbedrer tidligere studiers modeller og løsningsmetoder. Resultaterne sigter efter at støtte overgangen til en kollaborativ og bæredygtig logistik og bidrage til at kvantificere fordelene af integreret og kollaborativ optimering.

I anden del af afhandlingen adresserer vi passagertransport i byer. Den fjerde artikel omhandler on-demand mikrotransit, et transportsystem, der kombinerer effektiviteten af offentlig transport med fleksibiliteten af on-demand tjenester. Mens ride sharing-systemer er hurtigt voksende, hvilket bidrager til trængsel og socioøkonomiske ulighed, har den offentlige transport problemer med at tilpasse sig nye mobilitetsmønstre. Mikrotransit sigter mod at takle disse udfordringer ved at fremme effektive, bæredygtige og tilgængelig mobilitet. Vi præsenterer en forenet model til at designe netværket og optimere dets funktioner. Vi demonstrerer styrken af en ny modelleringstilgang i et virkelighedsnære case, og resultaterne validerer mikrotransit som et bæredygtigt og effektivt mobilitetssystem. Den sidste artikel fokuserer på jernbanetransport, hvor vi undersøger integreret optimering af køreplaner sammen med passagerruter. Vi præsenterer modellerings- og metodiske bidrag, og resultaterne fremhæver værdien af integration for passagerer og operatører.

Preface

This thesis was carried out at the Division of Management Science at DTU Management, Technical University of Denmark, in partial fulfillment of the thesis requirements for the degree of Doctor of Philosophy (Ph.D.) in Operations Research.

The project has been conducted between December 2019 and January 2023 under the supervision of Associate Professor Dario Pacino and Professor Stefan Røpke. The PhD studies also included a research visit to the *Sloan School of Management* at the Massachusetts Institute of Technology (MIT) in Cambridge, MA, United States, hosted by Professor Alexandre Jacquillat, between January and June 2022.

The PhD project has been supported by the Danish Maritime Fund under the *Collaborative Berth Planning in Liner Shipping (CoPlan)* project and by DTU Management.

This thesis consists of an introduction and five research chapters, four of which are based on academic papers that have been published, are currently under review or are expected to be submitted to an international peer-reviewed journal, and one is a short paper published in the proceedings of an international conference. The five paper-based chapters are co-authored, and they are each self-contained in terms of notation and with separate bibliographies.

Kgs. Lyngby, January 2023

Bernardo Martín-Iradi

Acknowledgements

First and foremost, I would like to thank my supervisors, Dario Pacino and Stefan Røpke for this three-year journey together. Dario Pacino, thanks for giving me the opportunity to pursue a PhD in Operations Research and for being a fantastic supervisor. Thanks for providing guidance and being supportive in all situations. It has been a pleasure both to share useful discussions and attend exotic conferences with you. Stefan Røpke, thanks for showing me the beauty of operations research already from my master's studies and inspiring me to pursue a career in academia. Thanks for always being open to discussing any ideas I had and always bringing fresh new light when I was stuck and everything seemed dark.

I want to deeply thank Alexandre Jacquillat for hosting me during my external stay at MIT. Thanks for your contagious excitement about research, and all the time and dedication. Our meetings were a true source of inspiration for me. I am also thankful to Kayla Cummings for being an incredible collaborator, for introducing me to the Operations Research Center (ORC) community, and for adding smartness and fun to all our meetings.

I also want to thank the PhD students at the ORC for welcoming me from day one, especially to Manuel Morán and Patricio Foncea for all the shared experiences and for making my stay unforgettable.

I am grateful to my colleagues in the Management Science division at DTU Management. To Harilaos Psaraftis for all the interesting and valuable talks and for being key in making my research stay possible. To Christina Scheel Persson for the unselfish help with all the paperwork. To Davide Cazzaro for being a true conference companion, and thanks also to my office-mates Alastair, Amandine, Baptiste, Gaspard and Sotiria for always being cheerful and keeping a positive atmosphere.

Thank you to Michael Lindahl and Joao Fonseca from Portchain for the interesting conversations that proved valuable in defining the scope of the problems.

I would like to thank DTU Management and the Danish Maritime Fund for supporting my PhD studies. I am also grateful to Otto Mønsteds Fond, Stibo-Fonden, and the Reinholdt W. Jorck og Hustrus Fond for funding my research stay abroad and attendance at a conference, which enhanced my PhD experience.

A special thanks to my "Padak" friends Ariadni, Sara, Shahana, and Ognjen,

for being a constant source of optimism and the perfect shelter when the days looked grey.

Thanks to my friends in Copenhagen, and my friends back in my hometown for their support and always welcoming me back as if time had not passed.

A huge thanks to Johanna for your continuous support during these years and for walking this journey by my side. Thanks for cheering me up through the highs and lows and always being willing to help.

Finally, I would like to thank my family for their endless support. Thanks for being a constant source of motivation and always encouraging me to try my best. You make me a better person, and I feel immensely privileged to have you by my side. Especially, I would like to dedicate this thesis to my parents. Thanks for everything.

To all of you: Thanks!

Contents

Part I

Introduction

Chapter 1

Introduction and thesis outline

Operations research (OR) is the discipline of applying advanced analytical methods to help make better decisions (INFORMS, 2012). OR is strongly tied to other scientific fields such as statistics, mathematical modeling and optimization, computer science, and engineering, and provides a wide range of problem-solving methods and techniques. These methods aim at improving the decision-making process for businesses, industry and society, and are present in almost every field, from marketing and finance to transportation and logistics.

The field of transportation and logistics is one of the principal fields where OR methods have been applied. The transportation of both freight and passengers is a core part of the proper functioning of the society. According to International Transport Forum (2021), passenger and freight transport demand is expected to more than double by 2050, which means that CO_2 emissions would triple the carbon budget to limit global warming at $1.5°C$. This trend requires highly ambitious policies to reduce emissions, and optimization techniques could have a major impact on them. The use of OR methods is present at all levels of transportation planning (Bektaş et al., 2019), from strategic planning, such as the design of transportation networks (Dukkanci et al., 2018), to tactical and operational vehicle routing and scheduling operations (Laporte, 2009). Advances in OR methods have provided significant benefits for the transport sector, and in combination with the digital evolution and new trends, they open exciting research opportunities.

Advances in the latest years in computational power and the development of novel and more efficient solution methods, have facilitated the study of more integrated operations and planning logistics. On the one hand, integrated optimization can provide economic benefits to planners. On the other hand, joint optimization of multiple logistics systems may require additional organizational changes or to engage in collaborative mechanisms (Expósito-Izquierdo et al., 2022). Therefore, it is relevant to study more in-depth and quantify the benefits of such integrated problems (Archetti and Speranza, 2016; Schiewe and Schöbel, 2022).

In this PhD thesis, we study transport optimization problems that arise from integrated operations and collaborative schemes. The main area of application is maritime seaside operations but we also study problems in urban road and railway transportation. We develop operations research methods to tackle

large-scale scenarios and that allows us to measure the potential benefits of collaborative and integrated problems in real-life settings that planners may encounter.

The remaining of this chapter is organized as follows. Section 1.1 provides a brief introduction to maritime logistics and urban transportation, the main two topics of this thesis. In Section 1.2, we introduce the context and objectives of this PhD project. Section 1.3 summarizes the scientific contributions of each chapter and how they have been disseminated. The conclusions are drawn in Section 1.4 and Section 1.5 discusses directions for further research.

1.1 Background and motivation

The main topic of this thesis is maritime transportation. However, the thesis also covers applications in urban (road and rail) transportation. This section aims at giving the reader some context about these two transportation areas and how they link to the research work of the thesis.

1.1.1 Maritime transportation

The introduction of containerization and the global adoption of standardized containers in the second half of the twentieth century had an enormous impact on the economies of freight transportation and became a key element of globalization (Hoovestal, 2013; Levinson, 2016) As Marc Levinson mentions in his book *The Box*, *"the container made shipping cheap, and by doing so changed the shape of the world economy"* (Levinson, 2016). Since then, maritime transportation has been growing steadily. More than 80 percent of global trade is carried by maritime transportation. In 2022, maritime trade accounted for more than 11 billion tons (UNCTAD, 2022), and it is expected to grow at an annual 2 percent in the upcoming 5 years.

Despite being one of the most efficient modes of freight transportation (International Chamber of Shipping, 2020), maritime transportation represents 3 percent of the total global greenhouse-gas emissions (IMO, 2020), and total shipping emissions have increased by 4.7 percent between 2020 and 2021 (UNCTAD, 2022). Although this growth is partly motivated by the recovery in maritime transport work after the COVID-19 pandemic, emissions have been steadily increasing during the last decade (see Figure 1.1). This trend conflicts with the ambitious strategy that the International Maritime Organization (IMO) adopted in 2018 for reducing greenhouse gas emissions from shipping by 50% in 2050 (IMO, 2018).

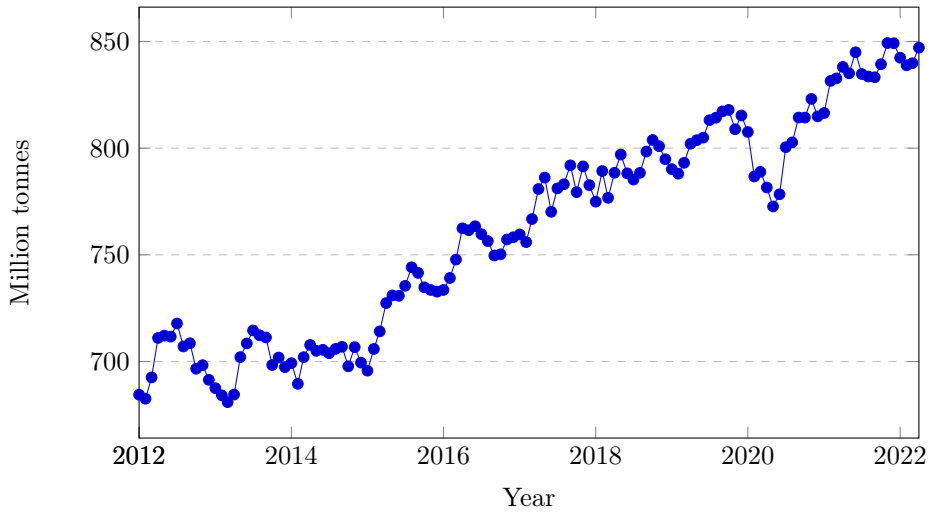


Figure 1.1: Total CO_2 emissions of world's merchant fleet. Source: UNCTAD (2022).

With the demand and fleet expected to continue increasing in the upcoming years (UNCTAD, 2022), shipping companies and port operators must find innovative ways of improving the efficiency of their operations. UNCTAD (2022) emphasizes the need to strengthen coordination across stakeholders by sharing information and developing solutions that lead to the overall best results. We believe that collaborative logistics involving shipping liner companies and terminal operators opens opportunities to improve operations and provide the industry with stronger planning tools.

Freight maritime transport is divided mainly into three modes of transport (Lawrence, 1972): (i) industrial shipping, (ii) tramp shipping, and (iii) liner shipping. In industrial shipping, the owners of the cargo also own the vessel fleet, and it is mostly used for the transportation of specific goods in large quantities to minimize costs. In tramp and liner shipping, the fleet owners offer transportation services to customers. In liner shipping, vessels sail on a fixed route following a published schedule. In tramp shipping, however, vessels do not follow a predefined route or schedule and operate as contract carriers where the cargo origin and destination are established upon agreement. This thesis deals with liner shipping operations, and next, we introduce liner shipping, and container terminals more in detail, focusing on their main logistical operations and how they interact.

Liner shipping

In terms of containerized cargo, 165 million 20-foot equivalent units (TEUs) were transported in 2021 by liner shipping companies (i.e., carriers). Most of these containers are transported using liner shipping networks. These networks are formed of multiple services or round trips. Each service visits a sequence of ports with a given frequency (e.g., weekly or biweekly), which defines a schedule. Each of the port visits is known as a port call, and is used by vessels to load and unload cargo. Figure 1.2 shows a global shipping network highlighting

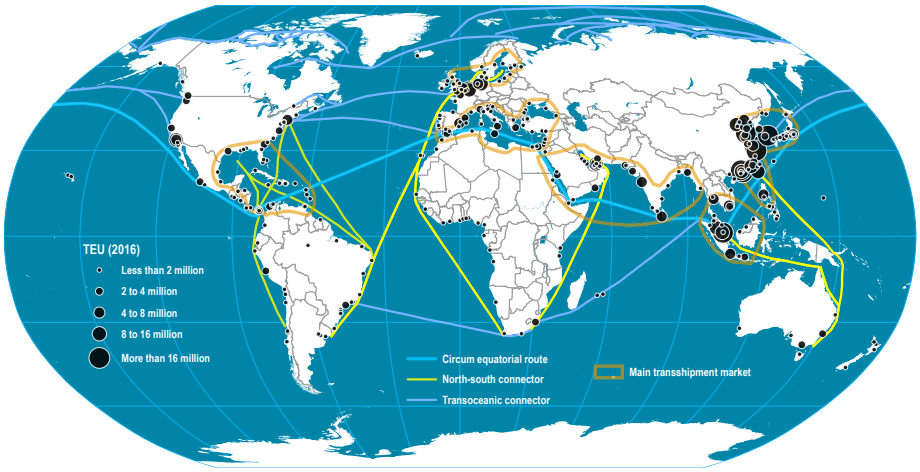


Figure 1.2: Global liner shipping network. Source: Notteboom et al. (2022).

the main corridors used by global services and the major ports in terms of container volumes, in TEUs. The total cost of operating a service for a carrier can be divided into multiple components. Fuel consumption constitutes the main source component, accounting for more than half of the total costs in some cases (Fagerholt and Psaraftis, 2015). Other costs include cargo handling costs, delay and port call fees, and canal passages.

Planning problems in liner shipping have been widely studied in the OR literature (see Christiansen et al., 2007, for a detailed overview of models and methods in maritime transportation). One of the main planning problems at the strategic level is the design of liner shipping networks, where the primary goal is to decide which liner routes to operate and at which frequency (Karsten et al., 2017; Christiansen et al., 2020). Tactical problems include, among others, the routing and scheduling of ships, where the decisions to be made involve defining the sequencing and exact visit times of the port visits in a route (Kjeldsen, 2011; Dulebenets et al., 2019), optimizing the sailing speed between ports (Psaraftis and Kontovas, 2015), and the allocation and routing of cargo, that

plans how to transport containers through the liner shipping network (Koza, 2019; Meng et al., 2014). Last, at the operational level, we find problems related to disruption management, where the goal is to find strategies to handle them, for example, deciding how to recover best the vessel's schedule (Brouer et al., 2013).

One of the most important tactical operations is the design of vessel schedules. This planning problem is known as the vessel scheduling problem (VSP), where the aim is to define the vessels' sailing speed between ports in a route to ensure an agreed service frequency. There are multiple objectives in designing a vessel schedule, and some of them are conflicting in essence. For instance, liner shipping companies want to minimize fuel consumption by sailing at low speeds, but they also want to minimize the number of vessels required to service a route at a given frequency. Liner shipping companies need to abide by additional restrictions such as arrival time windows at ports or maximum turnaround time. Efficient and reliable schedules are important not only for liner shipping companies but also for shippers and port operators, as they can help them make better planning of their operations. The VSP has been well studied in the literature (Meng et al., 2014; Lee and Song, 2017; Dulebenets et al., 2019). Most of the studies consider a cost-based objective where they minimize vessel fuel consumption, port handling, and late arrival costs (Fagerholt, 2001), or maximize the carriers' total profit based on the cargo delivered (Giovannini and Psaraftis, 2019). Fagerholt (2001) presented a mathematical model for the VSP, where vessels could arrive outside the agreed time window at the cost of a penalty. This is motivated by the fact that arriving outside the time window can result in significant cost savings. (Wang et al., 2014b) considered cargo allocation in the VSP and included costs of waiting at the port, and Dulebenets (2018) showed that negotiating port time windows and rates can lead to economical benefits.

The VSP is mostly optimized from the carrier's perspective. Moreover, many VSP studies consider a rather simple characterization of the handling times at port. These times are highly resource-dependent and can vary greatly if, for example, there is no space or equipment to serve the vessel. Integrating the VSP with container terminal operations can help characterize the operations more accurately. Chapters 2, 3, and 4 in this thesis address a hybrid problem that combines vessel scheduling with the allocation of ships to berthing positions in the quay. The chapters also include a more comprehensive literature review of the VSP.

Container terminals

Container vessels need to berth at dedicated container terminals where they can load and unload their containers. These terminals represent inter-modal

transport hubs where containers can be transferred between different ships but also to land transport modes such as trains or trucks.

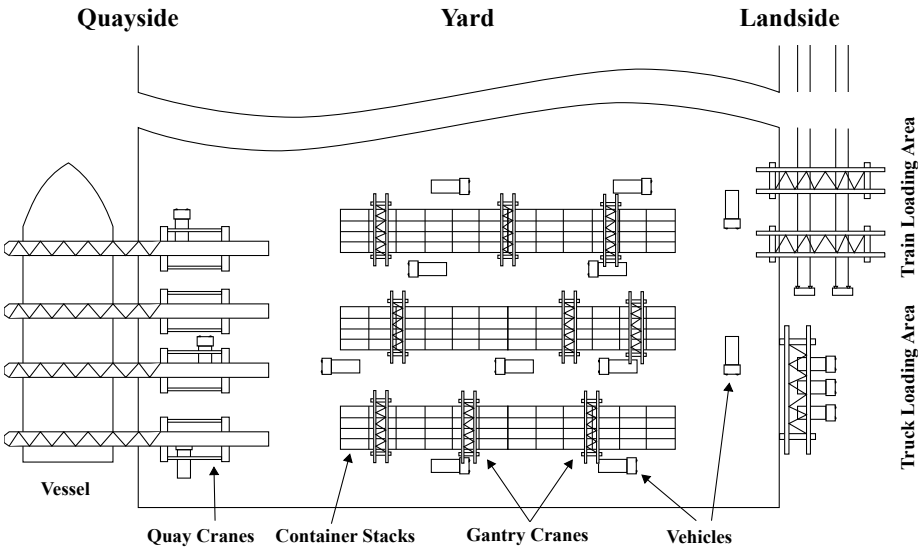


Figure 1.3: Container terminal layout. Source: Pacino and Jensen (2012).

As shown in Figure 1.3, each terminal is characterized by three main areas: (i) the quayside, (ii) the yard, and (iii) the landside. Both the quayside and landside can be seen as loading and unloading points, where the quayside or seaside is where ships are handled, and the landside defines the transfer point to land transport modes (i.e., trucks and trains). The yard is used as a temporary storage area for containers waiting to be loaded.

Container terminals involve many complex logistical operations (Carlo et al., 2014). Depending on the planning horizon, we can categorize the operations within the strategic (long-term), tactical (medium-term), and operational (short-term) levels. Figure 1.4 depicts an overview of the planning problems in container terminals, where problems are categorized into strategic, tactical, and operational levels depending on their planning horizon, and different operations are connected with arrows to indicate dependent planning components.

At the strategic levels, the planning problems mostly address the design of elements in the terminal such as designing the berthing (Vis and van Anholt, 2010) and yard area (Decastilho and Daganzo, 1993), or selecting which equipment and machinery to invest in (Vis, 2006).

Tactical problems address operations with a planning horizon of multiple weeks.

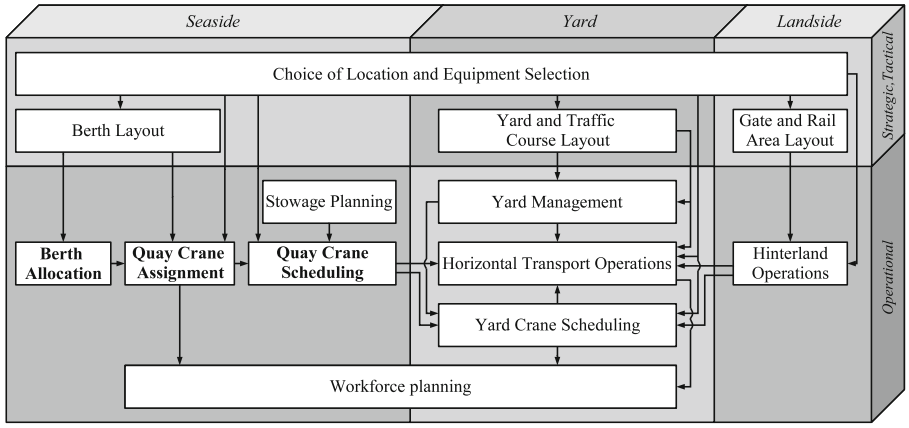


Figure 1.4: Planning problems in container terminals. Source: Meisel (2009).

Problems in this category include planning the fleet of the transport equipment within the terminal (Vis et al., 2005), and organizing the yard distribution (Cordeau et al., 2007) and berthing area (Fernández and Munoz-Marquez, 2022) based on the expected vessel services.

At the operational level, we find the majority of planning problems, generally aimed at maximizing resource utilization. On the quayside, the main decision problems involve planning the ships' berthing, assigning quay cranes to vessels and scheduling them (Daganzo, 1989; Meisel, 2009; Bierwirth and Meisel, 2015), or planning the storage configuration (i.e., stowage) of the ships (Monaco and Sammarra, 2008; Pacino and Jensen, 2012). Similarly, operations on the landside address scheduling of cranes and planning the loading and unloading of trucks and trains (Chen et al., 2013; Ambrosino et al., 2013). In the yard, the main operations involve the storage and stacking of containers (Stahlbock and Voß, 2008a) and transporting them between the yard and the quayside or landside (Stahlbock and Voß, 2008b).

One of the most critical seaside operations is planning berth allocations. This decision problem is known as the Berth Allocation Problem (BAP). The goal of the BAP is to assign arriving ships to berthing positions in the quay so that they can be serviced efficiently. The quay space is limited, which may lead to cases when ships may need to wait for a position to become available. Each ship has a desired time window to be serviced. Similarly, segments of the quay may have different operational time windows. There are two main variants of the BAP depending on the characterization of the quay. The discrete BAP divides the quay into a set of positions where each of them can be occupied by one ship

at a time. The continuous BAP, on the other hand, allows ships to berth at any point in the quay as long as they keep a safe distance from other ships. The BAP can also be categorized as static or dynamic. The static BAP assumes that the ships are already at the port, whereas the dynamic BAP considers an arrival time for each of the ships, which is defined a priori. Figure 1.5 depicts an

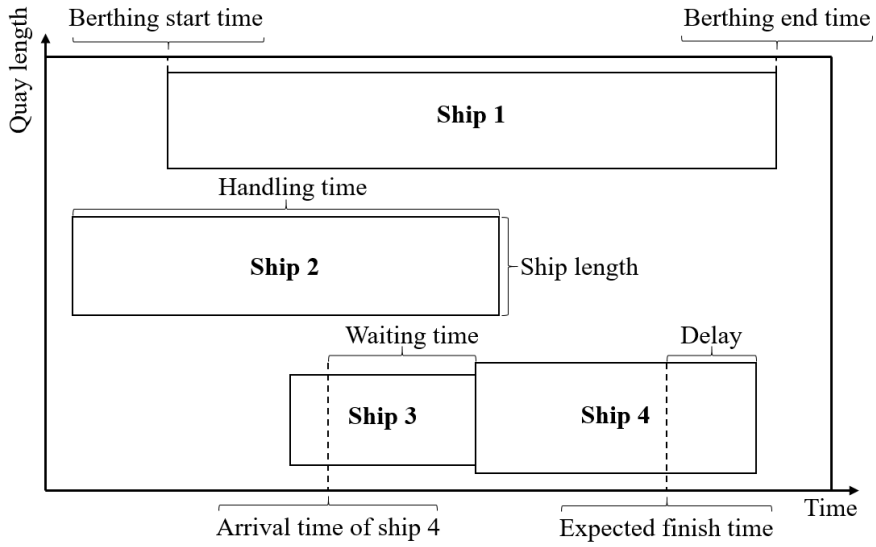


Figure 1.5: Example representation of the BAP with four ships.

example of the continuous and dynamic berth allocation in a two-dimensional diagram. The vertical axis illustrates the quay, and the horizontal axis, the planning horizon. Ships are represented as rectangles, with the handling time and ship length as their dimensions. The ships are subject to starting and finish times and need to be placed within the operating space without overlapping.

The BAP has received great attention in the literature during the last two decades (see Carlo et al., 2014; Bierwirth and Meisel, 2015, for surveys on the BAP). Lim (1998) introduced one of the first studies on the BAP and Imai et al. (2005) conducted one of the first studies considering a continuous BAP. Cordeau et al. (2005) studied both the discrete and continuous variants. There have been other studies considering other types of berths such as a hybrid version where each ship can berth in a subset of positions (Kordić et al., 2016), or indented berths where ships can be serviced from both sides (Beens and Ursavas, 2016). The BAP allows terminal operators to improve their operations by maximizing the utilization of the quay. Objectives of the BAP aim mainly at minimizing operational costs as a sum of the service time as well as waiting time at port and delays.

Container terminals do not have much interaction with other terminals and generally plan their operations independently. Due to multiple reasons, congestion in terminals does occur, impacting the schedules of the vessels involved. Vessels can easily propagate these delays to the next port calls they have scheduled, forcing terminals to modify their plans. We believe that more synchronized operations where the berth allocation of multiple ports is planned simultaneously can help improve the quality of the service at ports without negatively impacting the vessel services. In Chapter 2, 3, and 4 we study a multi-port BAP where the berth allocation is planned in multiple terminals simultaneously. Furthermore, Chapters 2 and 4 include a more comprehensive literature review on the BAP.

All of the terminal operations are interesting from an optimization perspective and have been addressed as decision problems using OR methods (Steenken et al., 2004). Container terminal planning problems are complex to solve and they are normally solved in a sequential approach, where the output of one problem is used as input for the next one. This approach significantly restricts the potential benefits of optimization. Nevertheless, as shown in Figure 1.4, some of the operations are tightly connected and are more amenable to integration. Studies that address integrated optimization of multiple operations, highlight the benefits of joint optimization. Some integrated problems consider operations involving different stakeholders, and may fall under the umbrella of collaborative logistics. In this thesis, we study collaborative problems arising from integrated maritime operations.

Collaborative and integrated maritime logistics

Considering all operations simultaneously quickly becomes intractable, and most of the shipping and terminal operations mentioned are solved in a sequential fashion. Some operations, however, are tightly related and if the integrated problem is not significantly complex, they can be optimized together. The main motivations for addressing integrated planning are the economic benefits and higher utilization of resources. Moreover, some of these operations share similar resources, which facilitates integration.

In the literature, we can find integrated studies both in liner shipping and terminal operations. Some studies integrating the liner shipping operations we mentioned include integrating network design and scheduling (Koza et al., 2020), service scheduling and cargo allocation (Wang et al., 2014a; Koza, 2019), ship routing and scheduling (Kjeldsen, 2011; Christiansen et al., 2013), or cargo allocation with speed optimization (Koza et al., 2020). Integrated container terminal problems are present in all three areas of the terminal. Examples of integrated optimization in seaside operations include combining the berth allocation and quay crane assignment (Meisel, 2009; Iris et al., 2015, 2017) to reduce the ship handling time. Some studies, also extended it to include crane

scheduling (Park and Kim, 2003; Rodrigues and Agra, 2022). Ultimately, there are some attempts to extend the berth allocation and quay crane assignment also to consider the yard assignment (Wang et al., 2018). Between the seaside and the yard, the scheduling of quay cranes is combined either with vehicle dispatch and scheduling operations (Tang et al., 2014), or with the management of the yard (He et al., 2015). The loading time highly depends on where in the yard the containers are. To address this, a common integration in the yard is to jointly optimize the yard allocation strategies with vehicle scheduling (Lee et al., 2009). Last, the scheduling of yard cranes can be optimized together with container storage policies (Jin et al., 2016).

The container fleet has constantly been increasing since 1990, and containerized trade is expected to grow at an annual 2.7 percent, faster than any other maritime trade segment (UNCTAD, 2022). Ports are struggling to absorb the increased demand, which is causing longer delays and more congestion in general. To mitigate this, companies need to look beyond short-term solutions for new and innovative strategies. Effective collaborative responses have the power to spread risks and adapt faster to disruptions. Initiatives such as the Clydebank Declaration (Gov.uk, 2021), an international pledge to foster partnership with all stakeholders to establish green shipping corridors, help support the transition towards a more sustainable and collaborative future in maritime logistics.

Despite the competitive environment, collaborative schemes exist in the maritime industry. The growth in vessel size and the need to utilize the capacity more efficiently led to some of the largest carriers seeking strategic alliances (Notteboom et al., 2022). These alliances can be seen as collaborative agreements between carriers that aim at consolidating operational synergies while also strengthening their market position. Currently, the three main alliances include the largest nine carriers and account for more than 80 percent of the global container shipping market (UNCTAD, 2022). Examples of other collaborative mechanisms in this context are sharing vessel space to meet the demand on given routes (i.e., vessel-sharing agreement), making capacity available for collaborating partners (i.e., slot-sharing), or sharing terminal capacity.

The container shipping industry has also been undergoing an important vertical consolidation process, resulting in carriers investing in container terminals. As a result, currently, the four largest carriers are among the top ten terminal operators (UNCTAD, 2022). These investments lead to carriers having more negotiation power and flexibility to plan their own operations. Carriers and terminal operators under the same ownership can facilitate the integration of liner shipping and terminal operations and enhance collaboration among carriers and terminal operators.

Studies based on collaborative problems are emerging significantly, motivated

by the benefits of joint optimization and the need for more resource-efficient logistics. In such problems, different stakeholders with different interests may be involved. The goal is to satisfy and provide gains to all of them to encourage participation and engagement in collaboration.

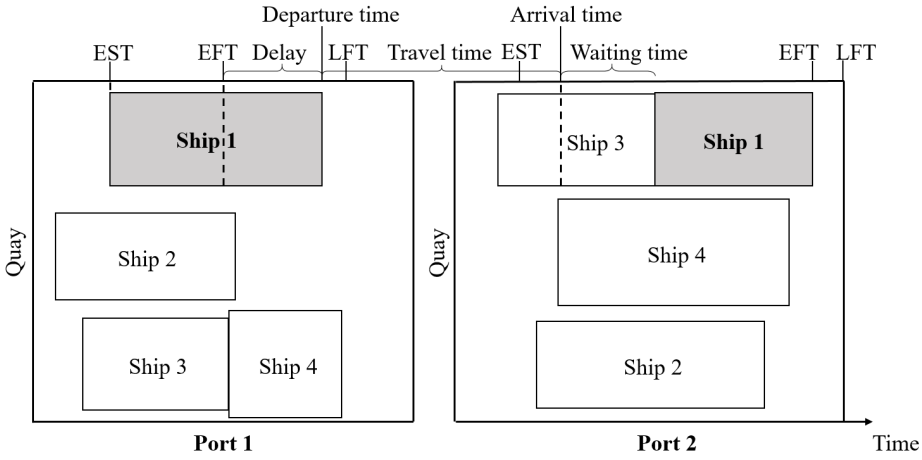


Figure 1.6: Example representation of the MBAP with four ships visiting two terminals. The timeline of operations for ship 1 is defined at the top, where EST, EFT and LFT denote the earliest start time, the expected finish time, and the latest finish time of the ship at the port.

Most collaborative problems between shipping companies and terminal operators arise in seaside operations. Problems such as the berth allocation or berth template design can be solved considering shipping liners' priorities (Wang et al., 2015; Ursavas, 2022). Furthermore, the berth allocation can be integrated with ship routing (Pang and Liu, 2014), and vessel scheduling (Venturini et al., 2017). The latter is known as the Multi-port Berth Allocation Problem (MBAP). The MBAP aims at simultaneously planning the berth allocation for a set of ships at multiple ports. The set of ships is characterized by sharing a subset of their port calls as part of their service. The problem extends the BAP to multiple ports, including vessel scheduling. For each ship, the arrival time at the port is conditioned by the departure time from the previous port and the selected sailing speed. The objective in general is to minimize the operating costs for both the ship owners and terminal operators, which includes waiting and handling costs, delay, and fuel consumption. Figure 1.6 shows an example of the MBAP considering two ports and four vessels. This problem enhances vessel scheduling by including accurate berth allocation planning, and it improves the BAP by including information from other terminals and optimizing accordingly. Chapters 2, 3, and 4 address the MBAP in detail both considering a continuous

and a discrete quay and measure the economic and environmental benefits of solving this collaborative problem.

1.1.2 Public and on-demand transportation

Public transportation is a passenger transport system aimed to provide mobility access to the general public. It is characterized by using high-capacity vehicles and operating along designated routes. The main goal of public transportation is to provide a viable and more sustainable alternative to private modes of transport, as it can help reduce congestion, travel times, and, thus, pollution. The socio-economical benefits are numerous. Having access to transportation grants people access to employment, medical assistance, and social opportunities. Its comparatively low price extends mobility access to people with scarce resources.

Public transportation does not only have an impact on users, investments in its development have proven to improve the system and impact surrounding businesses and communities (APTA, 2020). On the contrary, investments in road networks focused on private transport modes may not reduce traffic congestion and can actually make congestion worse if the improvements do not impact public transport positively. This is known as the Downs-Thomson paradox (Downs, 2005) and stems from Braess's paradox (Braess et al., 2005) that states that adding capacity to a road network can, in some cases, lead to a worse travel performance. As Lewis Mumford said: *"Adding highway lanes to deal with traffic congestion is like loosening your belt to cure obesity"* (Mumford, 1955). This strengthens the aim that future growth in the transport sector should primarily focus on public transport.

Despite the great benefits, public transport systems still face some major challenges and have failed to adapt to urban growth (Hazan et al., 2019). The advances in new technologies and poor reactions to changing commuting patterns have boosted the growth of alternative modes of transport. On-demand mobility or ride-sharing systems have been one of the most notable ones, operated by Transportation Network Companies (TNCs).

TNCs have been growing significantly. According to McKinsey & Co. (2021), the ride-sharing market has tripled in the period 2016-2019, reaching 40 million daily trips. This irruption, together with the global COVID-19 pandemic, has changed commuting patterns and has inevitably impacted public transportation, where ridership has decreased in many cities (The Economist, 2018). Ride-sharing systems provide a great customer experience, but their growth can have negative aspects. Diao et al. (2021) shows that ride-sharing systems contributed to a reduced public transport ridership, but an insignificant reduction in private car use. In fact, ride-sharing has contributed to congestion rather than alleviating

it (Erhardt et al., 2019). Lastly, ride-sharing systems have a higher fare than public transport services in general, which makes them inaccessible for some population segments.

A new transport system model that aims to tackle these challenges is *on-demand microtransit*.

Microtransit

Microtransit systems are defined as shared transportation systems that can offer both fixed-route schedules and flexible on-demand routing and scheduling (US Department of Transportation, 2016). This system aims to bring the digital advantages of ride-sharing into a public transportation context to expand access to affordable transportation. Microtransit leverages the benefits of public transport (e.g., high-capacity vehicles and advanced planning) and ride-sharing (e.g., on-demand service and planning flexibility) to provide a high-level service with low operational costs. Vehicles operate lines following a published schedule available for passengers. Trips need to be requested in advance, in general, similar to ride-sharing services. These requests are processed and optimized by the transport planner, resulting in the final route adjustments. Drivers receive the route updates, and passengers receive their trip details such as pick-up location and time, and estimated arrival time to destination. Pilot projects have been deployed across multiple cities with positive results (Eno Center for Transportation, 2018; Haglund et al., 2019). From the planning perspective, microtransit systems can be modeled by combining network design planning (i.e., deciding which lines to operate and their schedule) with vehicle routing operations (i.e., how to adjust the route deviations to the demand). The literature has mostly studied microtransit services from the operational point of view and at relatively small scales. Quadrifoglio et al. (2008) and Galarza Montenegro et al. (2022) study the routing deviations of individual lines by optimizing their pickup and dropoff locations. In Chapter 5, we address both the design and operation of microtransit systems by integrating the network design and on-demand routing operations in a single optimization problem.

1.2 Context

The goal of this PhD project is to explore optimization problems arising in transportation and logistics that leverage synergies and information-sharing between different stakeholders. This PhD thesis aims at developing advanced analytical tools to support collaborative maritime logistics and integrated optimization in other transportation areas, and demonstrate the benefits that integration and information sharing could report to stakeholders.

The research conducted in this PhD thesis contributes toward the Sustainable Development Goals (SDGs) (United Nations, 2015). All the scientific contributions have as one of the main targets to achieve more sustainable operations, either by reducing ship fuel consumption by sailing slowly between ports or by reducing vehicle distance in on-demand microtransit systems. This effort is categorized under *SDG number 13: Climate action*. One of the objectives of designing an on-demand microtransit system is to increase access to transport at affordable prices. This means also providing transportation services to a larger population segment and therefore, reducing inequalities. This objective is aligned with both *SDG number 10: Reduced inequalities* and *SDG number 11: Sustainable cities and communities*.

The thesis centers on the intersection between methodology and application, where we develop advanced solution methods envisioned to be applied in practice, and quantify the potential value for the decision-maker. The objectives of this thesis are the following:

- To study and formulate optimization problems based on logistical integration and collaboration. We study a collaborative problem that combines vessel scheduling and berth allocation planning. Two critical operations for ship-owners and terminals respectively.
- To develop algorithms to support the ongoing transition toward collaborative maritime logistics and integrated planning. We design large-scale optimization methods based on exact and heuristic methods to solve real-life instances. Exact methods exploit the structure of the problem using decomposition, and heuristic methods leverage problem knowledge to find high-quality solutions in short time.
- To analyze and quantify the value and benefits of collaborative and integrated problems in transportation. We conduct analyses to quantify the benefits of collaboration for the different stakeholders and present mechanisms to apply the problems in practice.
- To explore the applicability of integrated optimization in other transportation areas. We study the integration of train timetabling with passenger routing, and we address a rising mode of transport that integrates standard public transport and ride-sharing services.

This thesis is divided into four main parts. Part I contains the current chapter, and introduces the content of the thesis and outlines the remaining parts. Parts II, III, and IV are the core of the thesis and contain the research work conducted during the PhD project. Part II addresses the integration of vessel scheduling and berth allocation, two critical operations in maritime shipping.

This collaborative problem is based on information sharing between the stakeholders involved in the operations, shipping carriers and terminal operators. Part III studies a novel but rising mode of traffic, microtransit, which explores the integration of fixed public transport systems with on-demand ride-sharing systems. Last, Part IV studies the train timetabling operations both from the operator and passengers' perspective by integrating train timetable scheduling with passenger routing.

1.3 Scientific contributions

Part II consists of three chapters. Chapters 2 and 4 refer to academic journal papers while Chapter 3 is based on an international conference paper. Parts III and IV each consist of a single chapter based on academic journal papers. Two of the five chapters are published in international peer-reviewed journals, one chapter is published as a conference paper, one chapter is currently submitted to an international journal, and one chapter will be submitted to a journal in the next months.

In **Chapter 2, The multi-port berth allocation problem with speed optimization: Exact methods and a cooperative game analysis**, we address an emerging problem in collaborative maritime logistics, the multi-port berth allocation problem with speed optimization. This problem integrates vessel scheduling and berth planning operations, and therefore, requires terminal operators and shipping carriers to collaborate. The goals of this chapter are, (i) to present effective models based on problem reformulations that can solve large instances to optimality, and (ii) to study the potential benefits of the collaboration. We approach the problem by formulating the MBAP as a network-flow formulation. By applying decomposition techniques, we present a set partitioning formulation that is suitable for column generation techniques. Combining column generation with effective branching strategies, a novel set of valid inequalities, and additional enhancements, we present a *branch-and-cut-and-price* solution method. The method is tested in a set of benchmark instances from the literature and in an additional set of larger instances that we propose. The method proves to perform effectively and outperforms commercial solvers. Thanks to the tighter bounds of the reformulation, the algorithm converges in short computational times in most cases and achieves near-optimal solutions in the hardest cases.

We employ collaborative game theory to analyze and quantify the cost savings arising from the MBAP for carriers and terminal operators. By comparing the problem's collaborative setup to a non-collaborative scenario, we observe that all solutions maintain individual and group rationality. This means that from the carriers' or terminal operators' perspective, being part of the collabora-

tive scheme is the most attractive option. This encourages participation and strengthens the viability of such agreements. Finally, we describe alternative scenarios in which this type of problem could be applied. In the case of the participation of different companies, we envision third-party software companies acting as intermediaries. Many shipping and terminal operating companies already outsource their planning processes to these companies, which simplifies the process and does not require disclosing information between companies.

The work of Chapter 2 has been disseminated as follows:

- A journal paper co-authored with Dario Pacino, and Stefan Røpke published in *Transportation Science* (Martin-Iradi et al., 2022b).
- A presentation by Bernardo Martín-Iradi at the following virtual conferences:

  AIRO Young workshop 2021

  IFORS 2021

  ICCL 2021

  INFORMS Annual meeting 2021
- A presentation by Bernardo Martín-Iradi at EURO 2021, the *31st European Conference in Operational Research* in Athens, Greece.
- A seminar presentation by Bernardo Martín-Iradi held in 2020 at the Technical University of Denmark in Kgs. Lyngby, Denmark.

The berth allocation planning can be mathematically formulated in two ways depending on the spatial consideration of the quay. Either we define a finite set of positions that can host one ship each at a time, or we consider the entire quay and allow ships to occupy a part of it. In practice, the ships may vary greatly in length, and considering an independent set of positions may prove to be an ineffective use of the resources. In **Chapter 3, The multi-port continuous berth allocation problem with speed optimization**, we continue investigating the multi-port berth allocation problem with speed optimization, but in this case, the focus is on studying the impact of considering a continuous quay. Inspired by the model presented in Chapter 2, we present a network-flow formulation for the problem with a continuous quay. We exploit the same properties of the problem and apply decomposition techniques to define a set partitioning formulation. We combine column generation with a balanced branching strategy to develop a *branch-and-price* method. One of the consequences of modeling a continuous quay using a graph representation is that the size of the graph grows significantly. In our case, we define a segment length that establishes the minimum distance between two possible anchorage points. Based on the

ship's length, each ship occupies a different number of segments. This study also represents the first one addressing this problem, and therefore, we present an initial set of benchmark instances of moderate size, considering between 4 and 15 ships visiting 3 ports in the north of Europe. We compare the proposed *branch-and-price* method with commercial solvers solving the network-flow formulation. The computational results of the presented method outperform commercial solvers and provide tight near-optimal solutions. The practical insights indicate that considering a short segment length can provide planners with significant savings in operational costs, aligning with the premise that considering a continuous quay allows making more efficient use of the resources.

The work of Chapter 3 has been disseminated as follows:

- A peer-reviewed conference paper co-authored with Dario Pacino, and Stefan Røpke published in *Lecture Notes in Computer Science* (Martin-Iradi et al., 2022a).
- Presentation by Bernardo Martín-Iradi at ICCL 2022, the *13th International Conference on Computational Logistics* in Barcelona, Spain.
- A peer-review extended abstract and a presentation by Bernardo Martín-Iradi at TRISTAN 2022, the *11th Triennial Symposium on Transportation Analysis* in Mauritius.

Considering a continuous quay allows for better planning of the berthing operations, but the problems become computationally harder. We envision that the multi-port continuous berth allocation problem with speed optimization can be applied across large terminals and involve various carriers. This implies that the number of ships to optimize can be large. In **Chapter 4, An adaptive large neighborhood search heuristic for the multi-port continuous berth allocation problem**, we aim at solving large instances for the multi-port continuous berth allocation problem. For that, we present an adaptive large neighborhood search metaheuristic. The method iteratively generates new solutions by removing part of the solution and inserting the missing components in a different way. The method rewards the operators leading to better solutions, guiding the solution search process. We also present a mixed-integer problem formulation that can be solved by commercial solvers more efficiently than the network-flow formulation presented in Chapter 3. Based on real port data, we develop an instance generator and present a set of instances of large size. The heuristic method shows a robust and consistent performance achieving better solutions than commercial solvers in most cases. The method provides great scalability and can be further developed into a decision-support tool for planners.

The work of Chapter 4 has been disseminated as follows:

- A journal paper co-authored with Dario Pacino, and Stefan Røpke under review at *European Journal of Operational Research* (Martin-Iradi et al., 2023b).
- Presentation by Bernardo Martín-Iradi at EURO 2022, the *32nd European Conference in Operational Research* in Espoo, Finland. Initial version of the work with title: *An adaptive large neighborhood search for the multi-port continuous berth allocation problem with speed optimization.*

In **Chapter 5, Design and operation of on-demand microtransit systems**, we study an emerging urban transportation method, known as microtransit, that leverages the efficiencies of public transport and the flexibility of on-demand ride-sharing. We address the design and operation of this system by formulating it as a two-stage stochastic optimization problem, in which the first stage addresses strategic network design, and the second stage optimizes on-demand routing deviations to provide a better service to passengers. We model the second stage as a network flow problem using a subpath-based representation that scales better than equivalent segment-based or path-based representations. We present a solution method that combines Benders decomposition, to exploit the two-stage nature of the problem, and column generation, to effectively generate subpaths in the second stage. We demonstrate the scalability of the method by testing a case study in Manhattan. The method can solve instances with up to dozens of lines, hundreds of stops, and hundreds of requests. Last, we analyze the impact of microtransit by comparing it to standard ride-sharing and fixed-route transit. The results show that microtransit can increase coverage and level of service while maintaining low operational costs, which underscores the potential of microtransit toward sustainable mobility.

The work of Chapter 5 has been disseminated as follows:

- A working paper co-authored with Alexandre Jacquillat, and Kayla Cummings to be submitted as a journal paper to an international peer-reviewed journal (Martin-Iradi et al., 2023a).
- Presentation by Bernardo Martín-Iradi at INFORMS 2022, the *2022 INFORMS Annual meeting* in Indianapolis, United States.
- Presentation by Alexandre Jacquillat at the *Workshop on Transit Oriented Innovations in Emerging Mobility Service Designs, Algorithms, and Societal Implications* during the *2022 INFORMS*

Annual meeting in Indianapolis, United States.

- Seminar presentation by Bernardo Martín-Iradi held in 2022 at Technical University of Denmark in Kgs. Lyngby, Denmark.

In **Chapter 6, A column-generation-based matheuristic for periodic and symmetric train timetabling with integrated passenger routing**, we study railway operations for passenger transportation. To a large extent, the timetable planning process is still done manually by transit operators. Timetables are subject to a list of operational requirements, but they also need to comply with regulations and provide and adequate level of service to passengers. Passengers and transit planners have objectives that often are conflicting with each other. Operators want robust and cost-effective timetables, whereas passengers prefer high-frequency direct services to their destination. This creates a natural trade-off and highlights the importance of considering passenger travel information to design a good timetable. We address this by integrating passenger routing into the train timetable generation process. The timetable has multiple characteristics. In this case, we study the periodic and symmetric timetabling problem. Regional train systems usually operate using a periodic timetable and a symmetric one has the advantage of providing the same transfers between lines in both directions, which is a valued aspect by passengers. A graph representation of the problem enables to compute the schedule of a line in both directions by computing a single path in the graph. We solve the problem by combining column generation with separation techniques. To achieve feasible timetables we embed the method with a dive heuristic. The resulting matheuristic is denoted as a *dive-and-cut-and-price* method. To compute the passenger travel time, we route the passengers by solving a multi-commodity flow problem on a given feasible timetable. We use the passenger routing solution to derive a Benders optimality cut that is added to the original problem. The proposed algorithm is a large neighborhood search heuristic that iteratively solves the timetable generation and passenger routing procedures. We test the method in a regional and InterCity network in Denmark. The results indicate that the large neighborhood search achieves near-optimal solutions. These solutions are of high quality in line duration (i.e., reduced dwell time) and passenger travel time. The Benders' cut integration does not perform best when combined with the dive heuristic but experiments at the root node show that the cuts help tighten the optimality gap.

This study highlights the value of integrating operations where different stakeholders are involved, in this case, the transit operator and the passengers, and present efficient methods to automatize large parts of the timetable generation process and serve as a decision support tool for planners.

The work of Chapter 6 started as an MSc thesis project (Martin-Iradi, 2018) and, has been partly conducted before the start of the PhD project. The work has been disseminated as follows:

- A journal paper co-authored with Stefan Røpke published in *European Journal of Operational Research* (Martin-Iradi and Ropke, 2022).

1.4 Conclusions

The importance of applying optimization methods in transportation and logistics remains active. Recent advances in technology and digitalization enable new opportunities for OR applications. A growing research direction is to study the value of integrated optimization of multiple operations. These operations often involve stakeholders with different and sometimes opposing interests. In this thesis, we address this type of problem in three different transport sectors: (i) maritime logistics, (ii) on-demand road transit, and (iii) railway systems.

Part II focuses on collaborative logistics in maritime transportation. Although the container shipping industry is known to be a highly competitive one, collaboration agreements between different companies are becoming more frequent. We address seaside operations at the container terminals and study the MBAP that jointly optimizes vessel schedules and berthing plans. In Chapter 2, we present new formulations for the MBAP and design exact methods based on decomposition techniques. Results in large instances highlight the quality of the method and cooperative game theory shows that the MBAP can result in win-win scenarios, cost savings for carriers and terminal operators, and sustainable operations due to low sailing speeds. In Chapter 3 we extend the MBAP to consider a continuous quay. Results in real-port data indicate that a continuous quay can have a significant impact on operational costs. In Chapter 4, we address the same problem at a larger scale. We present a more amenable formulation for commercial solvers and develop a heuristic method. Results in large instances show the scalability and quality of the heuristic method.

Chapter 5 addresses an integrated approach between regular public transportation and on-demand mobility systems, denoted as microtransit. Microtransit aims to provide a high level of service to passengers at an affordable price point by combining the planning benefits of public transport with the flexibility of on-demand routing. The results indicate that microtransit systems can increase demand coverage while reducing operational costs and emissions. This type of system can help solve some of the challenges in urban transportation by strengthening the use of public transport and reducing congestion in cities.

Lastly, Chapter 6 looks at the integration of railway timetabling and passenger routing. We look at optimizing the timetable generation process, including the interests of passengers and the operator. The results show that timetabling and passenger routing can be jointly optimized, leading to better results than solving the two problems separately.

All in all, this thesis contributes toward more collaborative and integrated transportation. We have formulated optimization problems in maritime logistics based on collaboration and extended integrated optimization to other transportation areas. These problems exploit synergies between stakeholders and leverage information sharing to achieve improved and efficient planning decisions. Furthermore, all problems combine two or more complex optimization problems that require powerful algorithmic frameworks to solve them. We have developed advanced solution methods based on decomposition and heuristic techniques that, combined with novel modeling contributions, are capable of solving large instances in real-world setups for the first time. Moreover, these methods could be further developed to serve as decision-support tools for transport planners and practitioners. We have analyzed the practical implications of collaboration and logistical integration and measured the operational impact of the problems. The results show that collaborative and integrated optimization results in significant benefits for all the stakeholders. In the case of maritime logistics, these results are strengthened, showing that the most profitable scenario for each stakeholder happens when everyone collaborates. We believe these findings can encourage new carriers and transport operators to engage in collaboration. To conclude, we believe this thesis scientifically highlights the positive value of collaborative and integrated strategies and promotes them toward efficient and sustainable transportation.

1.5 Future work

During the PhD project, there have been multiple research directions that remained unexplored, but that may be worth considering in the future. We present four main ideas that we are currently researching or that we envision as future work.

1.5.1 Stochastic optimization in collaborative maritime logistics

Recent disruptions such as the COVID-19 pandemic or the war in Ukraine have impacted supply chains and increased costs worldwide. This highlights the need to protect and strengthen the shipping industry and has led shipping companies and ports to seek better ways to manage disruptions.

Inspired by the methodology and modeling perspective of the work in Chapter 5, we believe it can also be applied to optimize disruption management in maritime logistics. In Chapter 5, we use stochastic programming to account for uncertainty in passenger demand. This is done by defining a set of scenarios, each comprising a demand realization (e.g., a weekday morning rush-hour). We envision that a similar approach can be extended to collaborative problems in maritime logistics to manage disruptions. For the case of the multi-port berth allocation problem, the model can be reformulated as a two-stage stochastic problem. In the first stage, decisions about the vessel schedules are made. This involves determining the sailing speed between ports, but we could also include routing decisions such as skipping a port visit or altering the order of port visits (Brouer et al., 2013). The uncertainty is characterized by the nature of the disruptions where each realization (i.e., scenario) corresponds to a different disruption setting. Disruptions can be represented by a delayed ship, a congested or closed port, or an unexpected berth prioritization. The second-stage decisions would then involve adjusting the berth allocation plan in response to the disruption. Potentially, one could try to model the entire second stage as a BAP with uncertainty (Rodrigues and Agra, 2022) Last, the planning decisions could potentially be extended to consider transhipments.

1.5.2 Cargo routing and transhipments in collaborative maritime logistics

The liner shipping network is characterized by two main levels of distribution. On one level, we have the big container vessels that cover global routes and visit the largest ports, and on another level, we have the feeder vessels (i.e., smaller vessels) that transport the containers between the large ports and the smaller ones that are within a shorter distance. This means that a large number of containers need to be transhipped between ships at the terminal. In the case of the multi-port berth allocation problem, transhipments are not explicitly considered. Some examples of how to incorporate them are discussed in the conclusions of Chapter 2, but depending on the level of accuracy, it may require modeling changes. We differentiate two main approaches to model transhipments.

In the first approach, we define a set of predefined transhipments at each port. Each transhipment is characterized by a pair of ships, in which the first ship needs to unload the containers to be transhipped, and the second ship needs to load them. The requirement, in this case, is that the first ship must arrive before the second ship departs. In fact, the difference between the first ship's arrival and the second ship's departure must be larger than the handling time required to tranship the containers. To incorporate this approach in the multi-port berth allocation problem, we have two options. One one hand, we could restrict the port call duration of both ships and ensure that the latest finish time

of the incoming ship is strictly before the earliest starting time of the outgoing ship. On the other hand, we could add additional constraints in the problem formulation to guarantee the relative arrival of the incoming ship with respect to the departure of the outgoing ship. The first option is easier to implement and can be preprocessed, but it may be too restrictive. The second option provides more flexibility and potentially better solutions, but the problem becomes harder to solve.

In the second approach, we consider each container or group of containers as a commodity with an origin and destination port. Moreover, each commodity has a time window, with the earliest time to be loaded at the origin port and the latest time to be unloaded at the destination port. This can be formulated as a multi-commodity flow problem. This approach is more complex to embed in the multi-port berth allocation problem as it adds the cargo routing component to it. Inspired by Karsten et al. (2018), we could apply decomposition techniques to both the ship schedules and routes of the cargo. The authors combine Benders decomposition with column generation in a similar way as the algorithm we present in Chapter 5. In our case, we also need to consider the berth allocation at the terminals, which makes the overall problem an integration of three complex logistical operations. One could address this problem with a two-step approach like the one presented in Chapter 6 where the second stage solves a linear problem that allows generating efficient cuts to add to the main formulation. In our case, the first step would solve the multi-port berth allocation problem, and the second step would consider the routing problem as a linear programming problem following a similar procedure to Karsten et al. (2018).

1.5.3 Integration of quay crane assignment and scheduling

As mentioned in Section 1.1.1, the BAP can be naturally integrated with the quay crane assignment problem (QCAP). Integrating these two problems allows us to make more efficient use of the equipment and reduce handling time for ships. So far, the MBAP defines a fixed handling time for ships depending on the berthing position which may not be accurate. We believe that integrating the QCAP in the MBAP can help terminal operators to reduce their costs by improving their plan quality. Chapters 2 and 3 use a set partitioning formulation to solve the problem where the variables refer to entire schedules for ships including their berthing assignments. A similar formulation is used by Iris et al. (2015) to solve the integrated BAP and QCAP. where each variable represents a feasible assignment of a ship to a berth (i.e., position and time). In their case, they handle the crane assignment decision with a different variable which complicates the entire problem. A possible alternative is to embed the crane assignment in the variable itself. This makes the original problem less constrained, but the number of variables can grow significantly.

1.5.4 Dial-a-ride systems in on-demand microtransit systems

The work presented in chapter 5, addresses a version of the problem where all passengers have a common destination. This simplifies the logistics, but the problem remains complex, as decisions about where and when to pick up passengers need to be made. We believe it is important to address the transit operations more accurately and we are currently studying an extension of the problem that accommodates passenger drop-offs at different locations. We envision this can be integrated into the current structure of the problem while retaining tractability.

Another important aspect to further study is the composition of the fleet and how this can affect operations. We assume a homogeneous fleet of vehicles with the same capacity. In a practical setup, we do not expect all lines and frequencies to have a balanced demand. Therefore, it will be interesting to consider vehicle capacity when designing the network. Similarly, some lines may visit areas with more stable and localized demand and may not require the vehicle to do additional deviations. The opposite case, where demand is sparse and unlikely to be near the reference route, may require the vehicle to operate with more routing flexibility. Therefore, it is interesting to study if the network design could combine fixed-route bus lines with fully on-demand vehicles.

References

Ambrosino, D., Caballini, C., and Siri, S. (2013). A mathematical model to evaluate different train loading and stacking policies in a container terminal. *Maritime Economics and Logistics*, 15(3):292–308.

APTA (2020). Economic impact of public transportation investment. https://www.apta.com/wp-content/uploads/APTA-Economic-Impact-Public-Transit-2020.pdf. Accessed: 2023-01-27.

Archetti, C. and Speranza, M. G. (2016). The inventory routing problem: The value of integration. *International Transactions in Operational Research*, 23(3):393–407.

Beens, M. A. and Ursavas, E. (2016). Scheduling cranes at an indented berth. *European Journal of Operational Research*, 253(2):298–313.

Bektaş, T., Ehmke, J. F., Psaraftis, H. N., and Puchinger, J. (2019). The role of operational research in green freight transportation. *European Journal of Operational Research*, 274(3):807–823.

Bierwirth, C. and Meisel, F. (2015). A follow-up survey of berth allocation and quay crane scheduling problems in container terminals. *European Journal of Operational Research*, 244(3):12689, 675–689.

Braess, D., Nagurney, A., and Wakolbinger, T. (2005). On a paradox of traffic planning. *Transportation Science*, 39(4):446–450.

Brouer, B. D., Dirksen, J., Pisinger, D., Plum, C. E. M., and Vaaben, B. (2013). The vessel schedule recovery problem (vsrp) – a mip model for handling disruptions in liner shipping. *European Journal of Operational Research*, 224(2):362–374.

Carlo, H. J., Vis, I. F., and Roodbergen, K. J. (2014). Transport operations in container terminals: Literature overview, trends, research directions and classification scheme. *European Journal of Operational Research*, 236(1):1–13.

Chen, G., Govindan, K., and Yang, Z. (2013). Managing truck arrivals with time windows to alleviate gate congestion at container terminals. *International Journal of Production Economics*, 141(1):179–188.

Christiansen, M., Fagerholt, K., Nygreen, B., and Ronen, D. (2007). Chapter 4 maritime transportation. *Handbooks in Operations Research and Management Science*, 14(C):189–284.

Christiansen, M., Fagerholt, K., Nygreen, B., and Ronen, D. (2013). Ship routing and scheduling in the new millennium. *European Journal of Operational Research*, 228(3):467–483.

Christiansen, M., Hellsten, E. O., Pisinger, D., Sacramento, D., and Vilhelmsen, C. (2020). Liner shipping network design. *European Journal of Operational Research*, 286(1):1–20.

Cordeau, J. F., Gaudioso, M., Laporte, G., and Moccia, L. (2007). The service allocation problem at the gioia tauro maritime terminal. *European Journal of Operational Research*, 176(2):1167–1184.

Cordeau, J. F., Laporte, G., Legato, P., and Moccia, L. (2005). Models and tabu search heuristics for the berth-allocation problem. *Transportation Science*, 39(4):526–538.

Daganzo, C. F. (1989). The crane scheduling problem. *Transportation Research Part B-methodological*, 23(3):159–175.

Decastilho, B. and Daganzo, C. (1993). Handling strategies for import containers at marine terminals. *Transportation Research Part B-methodological*, 27(2):151–166.

Diao, M., Kong, H., and Zhao, J. (2021). Impacts of transportation network companies on urban mobility. *Nature Sustainability*, 4(6):494–500.

Downs, A. (2005). *Still stuck in traffic: Coping with peak-hour traffic congestion.* Brookings Institution Press.

Dukkanci, O., Bektaş, T., and Kara, B. Y. (2018). Green network design problems. *Sustainable Transportation and Smart Logistics: Decision-making Models and Solutions*, pages 169–206.

Dulebenets, M. A. (2018). A comprehensive multi-objective optimization model for the vessel scheduling problem in liner shipping. *International Journal of Production Economics*, 196:293–318.

Dulebenets, M. A., Pasha, J., Abioye, O. F., and Kavoosi, M. (2019). Vessel scheduling in liner shipping: a critical literature review and future research needs. *Flexible Services and Manufacturing Journal*, 33(1):43–106.

Eno Center for Transportation (2018). Uprouted: Exploring microtransit in the united states.

Erhardt, G. D., Roy, S., Cooper, D., Sana, B., Chen, M., and Castiglione, J. (2019). Do transportation network companies decrease or increase congestion? *Science Advances*, 5(5):eaau2670.

Expósito-Izquierdo, C., Expósito-Márquez, A., Melián-Batista, B., Moreno-Pérez, J. A., and Moreno-Vega, J. M. (2022). Intelligent collaborative freight distribution to reduce greenhouse gas emissions: A review. *Studies in Computational Intelligence*, 1036:133–142.

Fagerholt, K. (2001). Ship scheduling with soft time windows: An optimisation based approach. *European Journal of Operational Research*, 131(3):559–571.

Fagerholt, K. and Psaraftis, H. N. (2015). On two speed optimization problems for ships that sail in and out of emission control areas. *Transportation Research. Part D: Transport and Environment*, 39:56–64.

Fernández, E. and Munoz-Marquez, M. (2022). New formulations and solutions for the strategic berth template problem. *European Journal of Operational Research*, 298(1):99–117.

Galarza Montenegro, B. D., Sörensen, K., and Vansteenwegen, P. (2022). A column generation algorithm for the demand-responsive feeder service with mandatory and optional, clustered bus-stops. *Networks*.

Giovannini, M. and Psaraftis, H. N. (2019). The profit maximizing liner shipping problem with flexible frequencies: logistical and environmental considerations. *Flexible Services and Manufacturing Journal*, 31(3):567–597.

Gov.uk (2021). Cop 26: Clydebank declaration for green shipping corridors. https://www.gov.uk/government/publications/cop-26-clydebank-declaration-for-green-shipping-corridors/cop-26-clydebank-declaration-for-green-shipping-corridors. Accessed: 2023-01-26.

Haglund, N., Mladenović, M. N., Kujala, R., Weckström, C., and Saramäki, J. (2019). Where did kutsuplus drive us? ex post evaluation of on-demand micro-transit pilot in the helsinki capital region. *Research in Transportation Business and Management*, 32:100390.

Hazan, J., Lang, N. S., Wegscheider, A. K., and Fassenot, B. (2019). On-demand transit can unlock urban mobility. BCG Henderson Institute.

He, J., Huang, Y., Yan, W., and Wang, S. (2015). Integrated internal truck, yard crane and quay crane scheduling in a container terminal considering energy consumption. *Expert Systems With Applications*, 42(5):2464–2487.

Hoovestal, L. E. (2013). *The Economic Consequences of Containerization*, pages 55–72. Palgrave Macmillan US, New York.

Imai, A., Sun, X., Nishimura, E., and Papadimitriou, S. (2005). Berth allocation in a container port: using a continuous location space approach. *Transportation Research Part B-methodological*, 39(3):199–221.

IMO (2018). Initial IMO strategy on reduction of GHG emissions from ships. Technical Report MEPC.304(72), International Maritime Organization. (Accessed on 04.05.2020).

IMO (2020). Reduction of GHG emissions from ships. Fourth IMO GHG study 2020. Technical Report MEPC.304(72), International Maritime Organization. (Accessed on 22.08.2020).

INFORMS (2012). What is O.R.? https://www.informs.org/Explore/What-is-O.R.-Analytics/What-is-O.R. Accessed: 2023-01-06.

International Chamber of Shipping (2020). Environmental performance: Comparison of co2 emissions by different modes of transport. (Accessed on 22.01.2023).

International Transport Forum (2021). *ITF Transport Outlook 2021*.

Iris, C., Pacino, D., and Røpke, S. (2017). Improved formulations and an adaptive large neighborhood search heuristic for the integrated berth allocation and quay crane assignment problem. *Transportation Research. Part E: Logistics and Transportation Review*, 105:123–147.

Iris, C., Pacino, D., Røpke, S., and Larsen, A. (2015). Integrated berth allocation and quay crane assignment problem: Set partitioning models and computational results. *Transportation Research. Part E: Logistics and Transportation Review*, 81:75–97.

Jin, J. G., Lee, D. H., and Cao, J. X. (2016). Storage yard management in maritime container terminals. *Transportation Science*, 50(4):1300–1313.

Karsten, C. V., Brouer, B. D., Desaulniers, G., and Pisinger, D. (2017). Time constrained liner shipping network design. *Transportation Research. Part E: Logistics and Transportation Review*, 105:152–162.

Karsten, C. V., Røpke, S., and Pisinger, D. (2018). Simultaneous optimization of container ship sailing speed and container routing with transit time restrictions. *Transportation Science*, 52(4):739–1034.

Kjeldsen, K. H. (2011). Classification of ship routing and scheduling problems in liner shipping. *Infor*, 49(2):139–152.

Kordić, S., Davidović, T., Kovač, N., and Dragović, B. (2016). Combinatorial approach to exactly solving discrete and hybrid berth allocation problem. *Applied Mathematical Modelling*, 40(21-22):8952–8973.

Koza, D. F. (2019). Liner shipping service scheduling and cargo allocation. *European Journal of Operational Research*, 275(3):897–915.

Koza, D. F., Desaulniers, G., and Røpke, S. (2020). Integrated liner shipping network design and scheduling. *Transportation Science*, 54(2):512–533.

Laporte, G. (2009). Fifty years of vehicle routing. *Transportation Science*, 43(4):408–416.

Lawrence, S. (1972). *International Sea Transport: the Years Ahead*. Lexington books. Lexington Books.

Lee, C. Y. and Song, D. P. (2017). Ocean container transport in global supply chains: Overview and research opportunities. *Transportation Research Part B: Methodological*, 95:442–474.

Lee, D. H., Cao, J. X., Shi, Q., and Chen, J. H. (2009). A heuristic algorithm for yard truck scheduling and storage allocation problems. *Transportation Research Part E: Logistics and Transportation Review*, 45(5):810–820.

Levinson, M. (2016). *The Box: How the Shipping Container Made the World Smaller and the World Economy Bigger - Second Edition with a new chapter by the author*. Princeton University Press, rev - revised, 2 edition.

Lim, A. (1998). The berth planning problem. *Operations Research Letters*, 22(2-3):105–110.

Martin-Iradi, B. (2018). Optimization in railway timetabling for regional and intercity trains in Zealand. Master's thesis, Technical University of Denmark.

Martin-Iradi, B., Cummings, K., and Jacquillat, A. (2023a). Design and operation of on-demand microtransit systems. *Working paper*.

Martin-Iradi, B., Pacino, D., and Ropke, S. (2022a). The multi-port continuous berth allocation problem with speed optimization. In de Armas, J., Ramalhinho, H., and Voß, S., editors, *Computational Logistics*, pages 31–43, Cham. Springer International Publishing.

Martin-Iradi, B., Pacino, D., and Ropke, S. (2022b). The multiport berth allocation problem with speed optimization: Exact methods and a cooperative game analysis. *Transportation Science*, 56(4):972–999.

Martin-Iradi, B., Pacino, D., and Ropke, S. (2023b). An adaptive large neighborhood search heuristic for the multi-port continuous berth allocation problem. *Under review*.

Martin-Iradi, B. and Ropke, S. (2022). A column-generation-based matheuristic for periodic and symmetric train timetabling with integrated passenger routing. *European Journal of Operational Research*, 297(2):511–531.

McKinsey & Co. (2021). Shared mobility: Where it stands, where it's headed. https://www.mckinsey.com/industries/automotive-and-assembly/our-insights/shared-mobility-where-it-stands-where-its-headed.

Meisel, F. (2009). *Seaside operations planning in container terminals*. Physica-Verl.

Meng, Q., Wang, S., Andersson, H., and Thun, K. (2014). Containership routing and scheduling in liner shipping: Overview and future research directions. *Transportation Science*, 48(2):265–280.

Monaco, M. F. and Sammarra, M. (2008). The ship stowage planning problem. *International Conference on Harbour, Maritime and Multimodal Logistics Modelling and Simulation*, 1:208–213.

Mumford, L. (1955). The sky line. *The New Yorker, April 16th, 1955*.

Notteboom, T., Pallis, A., and Rodrigue, J. (2022). *Port Economics, Management and Policy*. Routledge.

Pacino, D. and Jensen, R. (2012). *Fast Generation of Container Vessel Stowage Plans: using mixed integer programming for optimal master planning and constraint based local search for slot planning*. PhD thesis. Pacino, Dario. "Fast Generation of Container Vessel Stowage Plans." (2012). PhD Thesis.

Pang, K. W. and Liu, J. (2014). An integrated model for ship routing with transshipment and berth allocation. *Iie Transactions (institute of Industrial Engineers)*, 46(12):1357–1370.

Park, Y. M. and Kim, K. H. (2003). A scheduling method for berth and quay cranes. *Or Spectrum*, 25(1):1–23.

Psaraftis, H. N. and Kontovas, C. A. (2015). Green maritime transportation: Speed and route optimization. *International Series in Operations Research and Management Science*, 226:299–349.

Quadrifoglio, L., Dessouky, M. M., and Ordóñez, F. (2008). Mobility allowance shuttle transit (mast) services: Mip formulation and strengthening with logic constraints. *European Journal of Operational Research*, 185(2):481–494.

Rodrigues, F. and Agra, A. (2022). Berth allocation and quay crane assignment/scheduling problem under uncertainty: A survey. *European Journal of Operational Research*, 303(2):501–524.

Schiewe, P. and Schöbel, A. (2022). Integrated optimization of sequential processes: General analysis and application to public transport. *Euro Journal on Transportation and Logistics*, 11:100073.

Stahlbock, R. and Voß, S. (2008a). Operations research at container terminals: A literature update. *Or Spectrum*, 30(1):1–52.

Stahlbock, R. and Voß, S. (2008b). Vehicle routing problems and container terminal operations - an update of research. *Operations Research/ Computer Science Interfaces Series*, 43:551–589.

Steenken, D., Voß, S., and Stahlbock, R. (2004). Container terminal operation and operations research - a classification and literature review. *Or Spectrum*, 26(1):3–49.

Tang, L., Zhao, J., and Liu, J. (2014). Modeling and solution of the joint quay crane and truck scheduling problem. *European Journal of Operational Research*, 236(3):978–990.

The Economist (2018). Public transport is in decline in many wealthy cities. www.economist.com/international/2018/06/21/public-transport-is-in-decline-in-many-wealthy-cities.

UNCTAD (2022). *Review of Maritime Transport 2022*. (Accessed on 16.01.2023).

United Nations (2015). Transforming our world: the 2030 agenda for sustainable development.

Ursavas, E. (2022). Priority control of berth allocation problem in container terminals. *Annals of Operations Research*, 317(2):805–824.

US Department of Transportation (2016). Shared mobility current practices and guiding principles. Technical report.

Venturini, G., Iris, C., Kontovas, C. A., and Larsen, A. (2017). The multi-port berth allocation problem with speed optimization and emission considerations. *Transportation Research. Part D: Transport and Environment*, 54:142–159.

Vis, I. F. (2006). A comparative analysis of storage and retrieval equipment at a container terminal. *International Journal of Production Economics*, 103(2):680–693.

Vis, I. F., de Koster, R. B., and Savelsbergh, M. W. (2005). Minimum vehicle fleet size under time-window constraints at a container terminal. *Transportation Science*, 39(2):249–260.

Vis, I. F. and van Anholt, R. G. (2010). Performance analysis of berth configurations at container terminals. *Or Spectrum*, 32(3):453–476.

Wang, H., Wang, S., and Meng, Q. (2014a). Simultaneous optimization of schedule coordination and cargo allocation for liner container shipping networks. *Transportation Research Part E: Logistics and Transportation Review*, 70(1):261–273.

Wang, K., Zhen, L., Wang, S., and Laporte, G. (2018). Column generation for the integrated berth allocation, quay crane assignment, and yard assignment problem. *Transportation Science*, 52(4):812–834.

Wang, S., Alharbi, A., and Davy, P. (2014b). Liner ship route schedule design with port time windows. *Transportation Research Part C: Emerging Technologies*, 41:1–17.

Wang, S., Liu, Z., and Qu, X. (2015). Collaborative mechanisms for berth allocation. *Advanced Engineering Informatics*, 29(3):572, 332–338.

Part II

Maritime logistics

Chapter 2

The multi-port berth allocation problem with speed optimization: Exact methods and a cooperative game analysis

Bernardo Martin-Iradi[a], Dario Pacino[a], and Stefan Ropke[a]

[a]DTU Management, Technical University of Denmark, 2800 Kongens Lyngby, Denmark

Status: Published in *Transportation Science*.

Abstract: We consider a variant of the berth allocation problem —i.e., the multi-port berth allocation problem—aimed at assigning berthing times and positions to vessels in container terminals. This variant involves optimizing vessel travel speeds between multiple ports, thereby exploiting the potentials of a collaboration between carriers (shipping lines) and terminal operators. Using a graph representation of the problem, we reformulate an existing mixed-integer problem into a generalized set partitioning problem, in which each variable refers to a sequence of feasible berths in the ports that the vessel visits. By integrating column generation and cut separation in a branch-and-cut-and-price procedure, our proposed method is able to outperform commercial solvers in a set of benchmark instances and adapt better to larger instances. In addition, we apply cooperative game theory methods to efficiently distribute the savings resulting from a potential collaboration and show that both carriers and terminal operators would benefit from collaborating.

Keywords: Transportation, Exact methods, Container terminal, Berth allocation problem, Speed optimization, Cooperative game theory

2.1 Introduction

The International Maritime Organization (IMO), in its fourth climate report (IMO, 2020), reflects on the increase in shipping's CO_2 emissions in the recent years. In the period 2012-2018 the shipping's total emissions have increased by 9.6%. This alarming trend highlights the need for pursuing the strategies that the IMO adopted in 2018 for reducing greenhouse gas (GHG) emissions from ships (IMO, 2018). The aim is to reduce total emissions from shipping by 50%

in 2050, and to reduce the average carbon intensity by 40% in 2030 and 70% in 2050, compared to 2008. Yet world maritime trade keeps growing at an annual average of 3% reaching a record high of 11 billion tons of total volume in 2018 —a number that translates into almost 800 million twenty-foot equivalent units (TEUs) handled in container ports worldwide (UNCTAD, 2019). Given that trade volume has steadily increased since then, the need for more efficient and sustainable operations in maritime transport logistics is essential (Bektaş et al., 2019).

From the terminal viewpoint, the growth in container trade involves more or larger vessels arriving at ports, in need of berthing. One solution to satisfying the increasing demand is to extend the existing quay. The problem is that doing so usually requires an expensive investment and sometimes may not even be physically feasible. An alternative strategy is to improve the efficiency of existing resources through optimization techniques that do not entail costly investment.

The berth planning of a terminal can be modelled mathematically as the Berth Allocation Problem (BAP). In the BAP, the aim is to assign incoming ships to berthing positions along the terminal. Steenken et al. (2004) define this problem as highly critical within container terminal planning logistics, due to the scarcity of berthing space. Figure 2.1 illustrates the problem in a two-

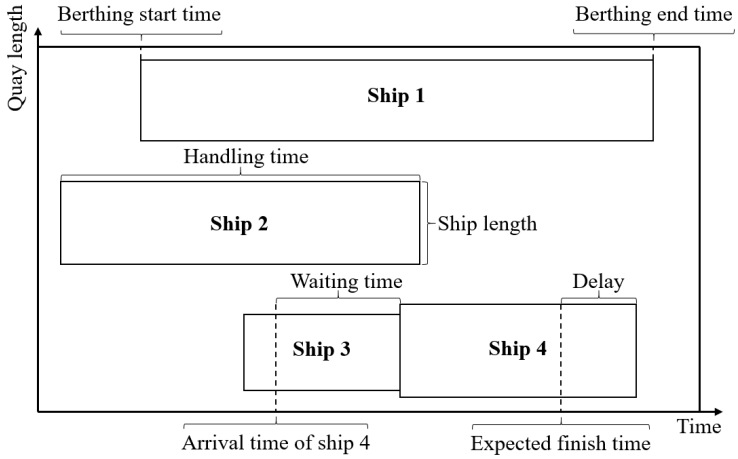


Figure 2.1: Example solution of the BAP for a port terminal with four vessels.

dimensional diagram where one dimension is space (quay length), and the other one is time (the planning horizon). We depict each ship as a rectangle whose dimensions are the ship length and handling time, the time the ship spends at the berth (i.e., during unloading and loading) Each ship usually has a fixed

time window defined by its expected start and finish time. Although ships can arrive before their berthing time, they will need to wait at the port. Similarly, ships can be allowed to exceed the expected finish time incurring in a delay. We denote the entire time that the ship spends at the port (i.e waiting time plus berthing period) as the "service time." Any non-overlapping positioning of the ship rectangles within the decision space defines a feasible solution for the BAP.

We can classify the BAP variants according to how the berths are distributed along the quay. In the *discrete* BAP, we divide the quay into a discrete set of berths, with only one ship allowed to one berth at a time, whereas in the *continuous* BAP, the ships can berth anywhere along the quay as long as they maintain a safe distance from the other ships. Moreover, the BAP can be either *static* or *dynamic*. In the *static* variant, we assume that all ships are at the port when the berth planning is done, whereas in the *dynamic* version, we assume that ships can arrive while the planning is in process.

The efficient planning of a terminal requires the vessels to abide by their schedules. Thus, efficient vessel scheduling is also a critical aspect, not only for the carriers, but also for the terminal operators. The design of vessel schedules can be modeled mathematically as the Vessel Scheduling Problem (VSP). The VSP aims at determining the sailing speeds between consecutive ports in the route (i.e., voyage legs) in order to optimize the vessels' fuel consumption and turnaround time at port and the number of vessels required to operate the route with a given frequency.

Both the increasing volume of container trade and the up-sizing of the vessels have led to increased competition among container terminals. each vying to become the port of call for more vessels (Notteboom et al., 2017). As a result, most terminals are reticent to share information with other terminals and prefer to plan their operations independently. Terminals commonly plan berth allocation based on ship schedules. Nevertheless, these schedules are subject to a level of uncertainty, because different types of disruptions—such as weather conditions or technical problems at the terminal—can alter the schedules and result in delays. When each terminal does its planning independently, a delay in one terminal can potentially be propagated through the shipping service to other ports (Notteboom and Vernimmen, 2009) or incur higher fuel costs for the carriers (shipping lines) if they need to increase the vessel's speed to make up for lost time. For example, a vessel stopping in ports A and B may encounter a congested terminal when arriving at port A and become delayed. The carrier can then order the vessel to either speed up to arrive at port B on time, entailing higher fuel consumption, or arrive late at port B, forcing the terminal to modify its berthing plan.

A potential solution to avoid this type of scenario is to establish some form of

collaboration between players in the maritime industry. Collaboration can be established not only between same type of stakeholders (i.e., between multiple carriers) but also between more players (i.e., carriers and terminal operators). The World Shipping Council (2015) encourages terminals to establish collaborative agreements with carriers as one of the main ways of reducing port congestion and improving planning efficiency.

A certain degree of collaboration is assumed in the VSP, however, the problem does not explicitly consider the berth allocation at the terminal and this can lead to a significant increase in service time. Therefore, integrating the BAP together with the scheduling of the vessels becomes relevant. Sharing information allows planners to simultaneously plan the berthing at the terminals and be able to minimize disruptions and reduce costs and emissions. Recent studies show that collaboration between carriers and terminals can lead to significant benefits for both (Dulebenets et al., 2019). This is the goal of the Multi-Port Berth Allocation Problem (MPBAP), first introduced by Venturini et al. (2017), which simultaneously plans the berth allocation of multiple ports taking into account the vessels' speed.

The MPBAP can be applied either when one company controls both vessels and terminals, or by a third-party service provider which works as an orchestrator. An example of the former is *Maersk*, owning both the carrier *Maersk Line* (Maersk, 2021) and the terminal operator *APM Terminals* (APM Terminals, 2021).

At present, there are companies in the market that offer optimization-based planning software separately to carriers and terminal operators (Portchain, 2021; Navis, 2021; Sealytix, 2021; TGI, 2021; RBS, 2021). Such companies already have access to all the necessary data for the MPBAP, which makes them excellent candidates to orchestrate the collaboration. Since both carriers and terminals are already sharing data with those companies, trust issues should be minimal, but customers should of course be free to decline that their data is used in a joint optimization problem. The amount of flexibility that terminals and carriers are willing to commit to the collaboration, can easily be modeled with the time windows, making the MPBAP if not an operational tool, at least a tool to identify the potential savings.

To make the service attractive to customers, the software company needs to show that the collaboration is beneficial for all involved parties. Therefore, we apply cooperative game theory to demonstrate that the total costs in the MPBAP solution can be shared in a favorable way. Using this service only requires that participating carriers and terminal operators allow the third party to jointly use their data but does not entail sharing additional data or the disclosure of the customer's data to other customers. Once the operations conclude, the

third party would be in charge of returning the savings according to the initial calculations.

Similar collaboration mechanisms have also been studied in the road transportation sector. Ergun et al. (2007) study collaborative logistics in truck transportation where part of the carriers' savings are returned to the shippers. Özener et al. (2011) also propose collaborative models where players receive more favorable rates in return. In fact, they indicate that a centralized decision-maker with complete information about all participants would be ideal for collaborative models to work. However, in their study, Özener et al. (2011) suggest that, due to lack of trust, players may not be willing to share additional information and therefore, they explore different mechanisms. Fortunately, this lack of trust is minimized in our case as the players already share the required information with the third party.

In studying the MPBAP, this paper makes four contributions. First, we present two new formulations for the MPBAP, based on a graph representation. Second, we propose exact methods based on column generation, together with branching, cutting, and symmetry-breaking enhancements. Third, we demonstrate the quality of our method by comparing it to a commercial solver and testing it through both a set of benchmark instances from a previous study and a new set of harder instances. Fourth, to demonstrate the benefits for both carriers and terminal operators in a scenario of a joint grand coalition, we apply cost allocation methods from cooperative game theory.

The structure of this paper is as follows. Section 2.2 reviews the state-of-the-art studies on berth allocation, speed optimization and collaboration on the shipping industry. Section 2.3 describes the MPBAP by presenting two mathematical formulations, together with the one from Venturini et al. (2017). Section 2.4 gives our solution method, and Section 2.5 introduces and discusses the cooperative game methods used for effectively distributing the costs of a coalition. Section 2.6 compares the models' performance through extensive computational experiments and analyzes the cooperative game theory results. Section 2.7 concludes by briefly discussing both the findings and possible future research directions.

2.2 Literature review

This section has been divided into three. First, we describe the main studies related to the BAP. Secondly, we cover the literature concerning speed optimization, and the last part focuses on collaboration studies within the container shipping industry and literature where cooperative game theory has been applied to it.

2.2.1 BAP literature

The berth allocation problem is known to be NP-hard (Lim, 1998; Hansen and Oguz, 2003) and has received significant attention in the last two decades. Carlo et al. (2014) and Bierwirth and Meisel (2015) presented detailed literature surveys on the seaside operations of container terminals such as the BAP where they emphasized the raising interest on this particular problem in the last years. Imai et al. (2005) conducted the first study considering a continuous BAP and Cordeau et al. (2005) studied both the discrete and continuous version of the problem and solved them through heuristic methods. Guan and Cheung (2005) presented a tree search exact method that performed better than commercial solvers and an efficient composite heuristic method. Du et al. (2015) extended the problem to also include the effect of tides and adopted the *virtual arrival* policy that is currently used in many terminals worldwide. Cheong et al. (2010) considered priorities for each of the vessels. The BAP is optimized using an evolutionary algorithm that minimizes the make-span, the waiting time and the deviation from a reference schedule. Buhrkal et al. (2011) compared three different methods for the discrete BAP and showed that a generalized set-partitioning model outperforms the rest. Saadaoui et al. (2015) reformulated the problem into a set packing problem where variables refer to assignments of ships to berthing positions and solved it using delayed column generation. In our paper, we combine the applicability of column generation procedures using a generalized set partitioning problem formulation. Regarding the discretization of the quay, Kordić et al. (2016) presented a hybrid variant of the BAP where ships can only berth in a given set of positions. Lalla-Ruiz et al. (2016a) studied how the tides can limit the time available for ships to berth given their draft and the water depth and solved this variant of the BAP using a generalized set partitioning problem formulation. The multi-port version of the BAP studied in this paper was first defined by Venturini et al. (2017). The mixed integer problem formulation they presented is used as a reference for the ones considered in this paper. Kramer et al. (2019) proposed two new formulations for the discrete BAP: a time-indexed formulation and an arc-flow formulation that seem to perform better than the methods from Buhrkal et al. (2011). Corry and Bierwirth (2019) proposed a mixed integer problem formulation for the BAP with channel-constrained ports where the sequencing of channel movements is also optimized.

2.2.2 Speed optimization literature

The relation between vessel speed and fuel consumption is non-linear. Since fuel emissions are directly proportional to the fuel burnt, optimizing sailing speed becomes relevant from the carrier and environmental perspective. The policies of the IMO in the last years have raised debate on which measures to implement regarding speed optimization, speed reduction or slow steaming. In

that aspect, multiple studies have been done analyzing the aspects and impacts of the different measures. Based on the scenario of slow steaming, Kontovas and Psaraftis (2011) investigated a berthing policy that aims at reducing the waiting time at port. Psaraftis and Kontovas (2013), Wang et al. (2013a), Psaraftis and Kontovas (2015a) and Psaraftis and Kontovas (2015b) presented taxonomies and surveys on speed models in the maritime transportation sector where the impacts and main trade-offs of slow steaming are analyzed and decision models proposed.

The VSP has speed optimization as its core concept and the interest in this problem has continued increasing in the last decade (Dulebenets et al., 2019). Fagerholt (2001) presented a mathematical model for the VSP and solved it using a method based on the set partitioning formulation. Wang et al. (2014) extended the VSP to also consider cargo allocation and indicated that carriers should consider the cargo costs arising from additional waiting time at port. Dulebenets (2018) proposed a multi-objective model considering the route service costs. The results indicated that negotiating the port calls and handling rates with the terminal operator could lead to significant savings. To some extent, the VSP can be seen as a collaborative problem, however, most of the studies focus on the interests of the carrier. A variant of the VSP where shipping line companies and terminal operators collaborate has also been studied recently. This variant assumes that the terminal operator can offer multiple time windows or handling rates to the carrier, instead of the fixed ones considered in the generic VSP. For instance, the MPBAP presented in Venturini et al. (2017) can be included in this problem category where the berth allocation planning is also considered. Dulebenets (2019) presented a mathematical model for the collaborative VSP where terminals offer both multiple port service time windows and handling rates. The results showed the benefits of the collaborative agreement on the liner shipping operations.

Environmental aspects have also been addressed in this type of problems. Fagerholt et al. (2010) minimized the fuel consumption by optimizing speeds along a shipping route. By discretizing the arrival times at each port, the cubic function relating speed and fuel emissions can be linearized and the problem solved as a shortest path problem. Fagerholt et al. (2015) and Zhen et al. (2020) extended the route and speed optimization study by also considering emission control areas (ECAs). Fagerholt et al. (2015) aimed at minimizing the fuel consumption whereas Zhen et al. (2020) also considered SO_2 emissions. Both studies showed that carriers tend to use slow steaming within ECAs or directly avoid sailing through these areas. Reinhardt et al. (2016) optimized a liner shipping network by adjusting berthing times with the objective of minimizing fuel consumption. The speed and routing of multiple vessels is optimized in Wen et al. (2017) under a unified objective that minimizes transit times, total costs and fuel emissions. They implemented a branch-and-price heuristic and a constraint programming

model which is tested in a subset of the Mediterranean ports. Du et al. (2011), Du et al. (2015) and Sun et al. (2018) integrated speed optimization with the BAP by considering that ships still need to sail a certain distance to arrive at port. The second-order cone programming transformation used by Du et al. (2011) to approximate the relation between sailing speed and fuel consumption is improved by quadratic outer approximations in Wang et al. (2013b).

2.2.3 Collaboration in the shipping industry

The MPBAP introduced by Venturini et al. (2017) can be seen as a problem with a high degree of collaboration and the study of different collaborative forms in the container shipping industry has gained interest in recent years. Song (2003) studied competition and co-operation in ports and coined the term *co-opetition*. Wang et al. (2015) presented two collaborative methods between shipping line companies and port operators where the aim is to create a win-win situation by balancing the priorities of both parties and encouraging them to share true information. Lalla-Ruiz et al. (2016b) proposed a cooperative search for the discrete BAP based on a grouping strategy. Individuals are organized into groups where they can only share information with other individuals from the same group. Notteboom et al. (2017) investigated alliance formations in container shipping by studying their strategies when choosing ports. Dulebenets et al. (2018) presented the collaborative berth allocation problem (CBAP), which is a variation of the BAP that also allows to divert vessels to another terminal when there is a peak demand, and solved it using a memetic algorithm. Collaboration is also studied by integrating berth allocation with other scheduling problems such as ship routing. Pang and Liu (2014) studied such integration for a feeder company operating both vessels and container terminals. This study also considered transhipments of containers but did not cover speed optimization.

Game theory has also been widely applied in the container shipping industry (Pujats et al., 2020). In our paper, the focus is on cooperative game theory where the target is on distributing the profits or savings among players. The studies vary depending on which are the players considered (carriers, terminal operators or both) in the cooperation. Song and Panayides (2002) applied cooperative game theory to depict a conceptual framework for liner shipping alliances showing that the core theory is applicable to the liner shipping market. Saeed and Larsen (2010) presented a two-stage cooperative game for container terminals within the Karachi Port in Pakistan. The results indicated that a *grand coalition* among all players gives the best payoff for all terminals. The work by Krajewska et al. (2008) showed, by means of cooperative game theory, that collaboration among road freight carriers is practical and cost-effective for all players. Wen et al. (2019) studied the benefits of horizontal cooperation in a shipping pool by not only maximizing the pool profit but also allocating the profits fairly among participants. The profit sharing framework from Krajewska

et al. (2008) and some of the profit allocation methods presented in Wen et al. (2019) have been used in this study and, to the best of our knowledge, it is the first time cooperative game theory is applied to the MPBAP.

2.2.4 Research gap

While the BAP and VSP have been extensively studied in the literature, with an increasing interest in the last decade, very few papers address the potentials of integrating the two problems and only one paper has been found to address the MPBAP. Only an MIP formulation for the problem has been proposed, which shows good performance for small instances but struggles when the size of the instances increases. Therefore, there is a need for a more efficient solution method that scales better to larger instances. Furthermore, the MPBAP implies collaboration between different parties in the shipping industry and an analysis of the model's applicability in real life is lacking in the literature. Thus, we believe that assessing the stakeholders' incentives to enter into such collaboration is relevant.

2.3 Problem description

The MPBAP can be seen as a partial integration between the BAP and the VSP. Particularly, this study is based on the discrete and dynamic BAP and it is extended to cover multiple ports where the sailing speed between ports is optimized. This can be seen as a collaborative approach where information is shared among shipping line and terminal companies. The main addition of the MPBAP compared to the BAP is the optimization of the sailing speed between ports and the simultaneous planning of multiple ports. Figure 2.2 shows a solution example to a problem with four ships and two ports, each having three berthing positions. As shown for ship 1, the travel time, which depends on the chosen sailing speed, determines the arrival time to the next port and this can constrain the available berthing time window further. The MPBAP aims at minimizing the total costs for both the carriers and terminal operators.

2.3.1 Fuel consumption model

One of the main costs for a carrier is the fuel. The fuel consumption is directly linked to the sailing speed but not in a linear way. Thus, we need an accurate model that links the sailing speed with the fuel consumption realistically.

Many studies approximate the fuel consumption as a cubic function of the speed (e.g., Meng and Wang (2011), Wang and Meng (2012), Reinhardt et al. (2016)),

$$F(i,\delta) = \left(\frac{\delta}{\delta_i}\right)^3 \Gamma_i \tag{2.1}$$

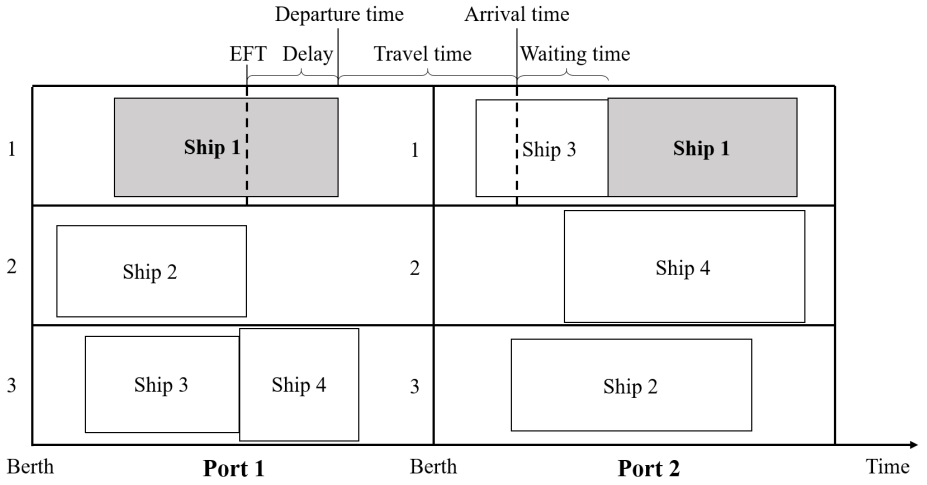


Figure 2.2: Example solution of the MPBAP for four vessels, two port terminals and three berths per port. The traveling timeline for ship 1 (in gray) is defined at the top, where EFT denotes the *expected finish time* at port 1.

where equation (2.1) measures the fuel consumption $F(i, \delta)$ in tons/hour for a given ship i. δ_i is the design speed of vessel i and δ is the sailing speed, both measured in knots (i.e., nautical miles per hour). Finally, Γ_i is the fuel consumption in tons/hour for vessel i at the design speed. This approximation is fairly accurate for container ships of limited size and for a range of sailing speeds that are not significantly slow. In our study, we optimize the sailing speed between ports, where we expect speeds similar to the design speed (δ_i) of the vessel and we do not consider the fuel consumption derived from entering or leaving a port where near-zero speeds are used. In order to avoid non-linearity in the mathematical formulation of the problem, we apply a discretization of the cubic approximation based on the one proposed by Venturini et al. (2017). A set of different speeds S is defined that can be used by ships to travel between ports. The set of speeds correspond to reasonable and realistic speeds in a range around the design speed. Then, for each of the selected speeds $\delta \in S$ and ship i, a fuel consumption value ($\gamma_{i,\delta}$) measured in tons/nautical mile can be calculated based on the cubic approximation using the following equation (2.2).

$$\gamma_{i,\delta} = \frac{F(i,\delta)}{\delta} = \frac{\left(\frac{\delta}{\delta_i}\right)^3 \Gamma_i}{\delta} \tag{2.2}$$

2.3.2 Cost structure

The MPBAP aims at optimizing the operational costs for both carriers and terminal operators. This Section defines the main costs involved in the problem context and describes to which stakeholder the costs are related. An overview

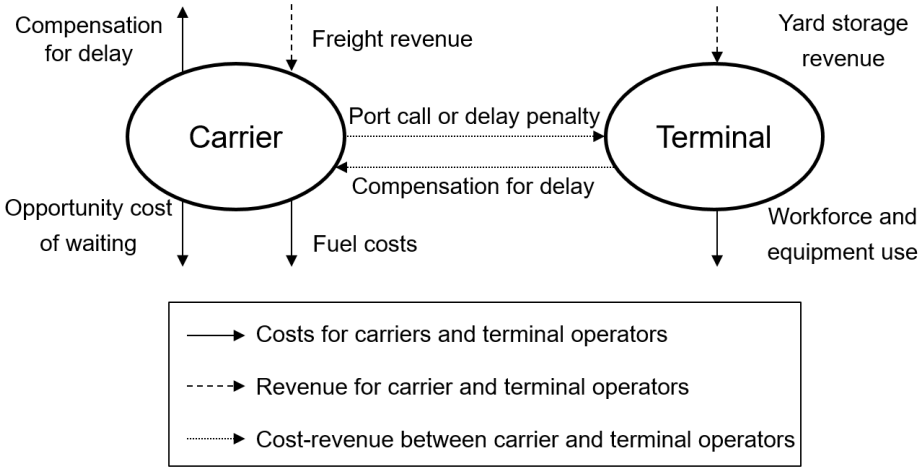


Figure 2.3: General overview of main costs and revenue for the shipping carriers and terminal operators.

of the main sources of cost and revenue for both carriers and terminal operators is shown in Figure 2.3.

As mentioned in Section 2.3.1, the main cost driver for a shipping line company is the fuel consumption which usually accounts for more than 50% of the carrier's total costs (Fagerholt and Psaraftis, 2015). Another carrier related cost is the waiting time at anchorage (i.e., waiting to berth at port). As described by Chang et al. (2012), the waiting cost is not only the direct cost of being for longer time at a port, but also the resulting loss of potential income (i.e., opportunity cost). Regarding the service time at port, this is usually pre-established by a contract or when booking the port call. The cost may differ based on multiple factors such as the number for containers to be loaded and unloaded (i.e., quay crane moves) or the size of the ship and number of cranes required. In this case, the cost can be considered constant for the carrier regardless of the resulting quality of the terminal's planning. Finally, there are also delay costs associated to the carrier. Ending the service after the expected finish time at a port may result in additional payments to the shippers for the delay on the delivery of their cargo. In addition there may be other costs arising from delays. On one hand, if the delay at the terminal is caused by the ship arriving late, the carrier may be subject to a fine or delay penalty to the terminal. On the other hand, if the

planned service time for a ship gets extended due to, for example, a breakdown of a quay crane or a poor berth allocation plan, the carrier affected by the delay may be entitled to a compensation from the terminal operator. It can be noticed, that these delay costs are paid from the carrier to the terminal operator or vice versa. This means that a cost for one party becomes a revenue for the other one. The main premise of this problem is that carriers and terminal operators jointly plan their operations and, therefore, the internal delay costs do not exist and can be excluded from the objective of the problem.

The main costs impacting the planning of the terminal in this problem are both the handling and delay costs. We identify the use of resources to be directly proportional to the number of workforce and quay crane usage hours. The fewer shifts needed to serve the vessel, the greater the profit is for the terminal. Therefore, both an increasing handling time by the vessels or an increased delay will require additional workforce. As mentioned before in this section, one of the premises of the MPBAP is that the planning decisions are agreed between the carrier and the terminal operator based on the overall best solution for all. This collaborative optimization removes the concept of delay between the participating players. However, we do consider a delay cost for the terminal operators in the objective of the problem. We study the problem from a tactical point of view but assume that, for instance, workforce planning at the terminal is performed beforehand. In this scenario, the suggested optimal berth allocation plan for a given terminal may require more workforce than initially planned. This will directly translate in the use of additional resources to cover for the additional handling operations that can be computed as delay costs.

All in all, the MPBAP covers the costs depicted with a continuous line in Figure 2.3. Thus, the objective of the problem focuses on minimizing the fuel consumption and the costs related to waiting, handling and delay time.

2.3.3 Mixed-integer problem formulation. The Venturini et al. (2017) model

The solution method presented in this paper is based on a mixed integer problem (MIP) formulation from Venturini et al. (2017), which we now briefly present. We first list the notation used in the model:

Sets and parameters	
N	Set of ships
P	Set of ports
P_i	Set of ports to be visited by ship $i \in N$ sorted in visiting order
B_p	Set of berths at port $p \in P$

$V^{p,b}$	Set of vertices, $V^{p,b} = N \cup \{o(p,b), d(p,b)\}$, with $o(p,b)$ = origin node for arcs and $d(p,b)$ = destination node for arcs, both defined for every port $p \in P$ and berth $b \in B_p$
$A^{p,b}$	Set of arcs (i,j) with $i,j \in V^{p,b}, i \neq j$
S	Set of speeds
$Start_i^p$	Minimum starting time of activities for ship $i \in N$ at port $p \in P_i$
EFT_i^p	Expected finishing time of activities for ship $i \in N$ at port $p \in P_i$
$s^{p,b}$	Starting time of activities for berth $b \in B_p$ at port $p \in P$
$e^{p,b}$	Ending time of activities for berth $b \in B_p$ at port $p \in P$
$h_i^{p,b}$	Handling time of ship $i \in N$ at berth $b \in B_p$ at port $p \in P$
$d^{p,p'}$	Distance between pair of ports $p, p' \in P$
P_{iL}	The last port to be visited by ship $i \in N$ in the route
$\gamma_{i,\delta}$	Fuel consumption per unit of distance for ship $i \in N$ at speed $\delta \in S$
Δ_δ	Travelling time per unit of distance when travelling at speed $\delta \in S$
$M1^{p,b}$	Big-M value, $M1^{p,b} = e^{p,b}$
$M2_i^{p,b}$	Big-M value, $M2_i^{p,b} = e^{p,b} - h_i^{p,b}$
F_c	Fuel consumption cost in \$ per ton
H_c	Handling activities cost in \$ per hour
I_c	Idleness cost in \$ per hour
D_c	Delay cost in \$ per hour

Decision variables

$y_{i,j}^{p,b} \in \mathbb{B}$	1 if ship j immediately succeeds ship i at berth $b \in B_p$ at port $p \in P$ where $(i,j) \in V^{p,b}$; 0 otherwise
$v_{i,\delta}^p \in \mathbb{B}$	1 if ship $i \in N$ sails from port p to some other port $p'(p, p' \in P_i := p \prec p')$ at speed $\delta \in S$; 0 otherwise
$T_i^{p,b} \in \mathbb{Z}^+$	Time at which ship $i \in N$ berths at berth $b \in B_p$ at port $p \in P_i$ (berthing time)
$T_{o(p,b)}^{p,b} \in \mathbb{Z}^+$	Time at which berth $b \in B_p$ at port $p \in P_i$ starts berthing ships (i.e., arrival time of the first ship to the berth)
$T_{d(p,b)}^{p,b} \in \mathbb{Z}^+$	Time at which berth $b \in B_p$ at port $p \in P_i$ finishes berthing ships (i.e., departure time of the last ship from the berth)
$T_i^p \in \mathbb{Z}^+$	Time at which port $p \in P_i$ opens activities for ship $i \in N$
$\Delta EFT_i^p \in \mathbb{Z}^+$	Difference between effective finishing time and EFT_i^p for ship $i \in N$ at port $p \in P_i$

The mathematical model is presented below:

$$\min \sum_{i \in N} \sum_{p,p' \in P_i \setminus \{P_{iL}\}: \{p \prec p'\}} I_c \left(\sum_{b \in B_{p'}} T_i^{p',b} - \sum_{b \in B_p} T_i^{p,b} + \sum_{b \in B_p} h_i^{p,b} \Big(\sum_{j \in N \cup \{d(p,b)\}} y_{i,j}^{p,b} \Big) - \sum_{\delta \in S} \Delta_\delta d^{p,p'} v_{i,\delta}^p \right)$$
$$+ \sum_{i \in N} \sum_{p \in P_i} \sum_{b \in B_p} H_c \Big(h_i^{p,b} \sum_{j \in N \cup \{d(p,b)\}} y_{i,j}^{p,b} \Big) + \sum_{i \in N} \sum_{p \in P_i} D_c \Delta EFT_i^p + \sum_{i \in N} \sum_{p,p' \in P_i \setminus \{P_{iL}\}: \{p \prec p'\}} \sum_{\delta \in S} F_c (\gamma_{i,\delta} d^{p,p'} v_{i,\delta}^p) \tag{2.3}$$

subject to:

$$\sum_{b\in B_p}\sum_{j\in N\cup\{d(p,b)\}} y_{i,j}^{p,b} = 1 \quad \forall i \in N, p \in P_i \tag{2.4}$$

$$\sum_{j\in N\cup\{d(p,b)\}} y_{o(p,b),j}^{p,b} = 1 \quad \forall p \in P, b \in B_p \tag{2.5}$$

$$\sum_{j\in N\cup\{o(p,b)\}} y_{j,d(p,b)}^{p,b} = 1 \quad \forall p \in P, b \in B_p \tag{2.6}$$

$$\sum_{j\in N\cup\{d(p,b)\}} y_{i,j}^{p,b} - \sum_{j\in N\cup\{o(p,b)\}} y_{j,i}^{p,b} = 0 \quad \forall i \in N, p \in P_i, b \in B_p \tag{2.7}$$

$$T_i^{p,b} + h_i^{p,b} - \left(1 - y_{i,j}^{p,b}\right) M1^{p,b} \leqslant T_j^{p,b} \ \forall (i,j) \in A^{p,b}, p \in \{P_i \cap P_j\}, b \in B_p \tag{2.8}$$

$$\sum_{b\in B_p} T_i^{p,b} + \sum_{b\in B_p} h_i^{p,b}\Big(\sum_{j\in N\cup\{d(p,b)\}} y_{i,j}^{p,b}\Big) + \sum_{\delta\in S} \Delta_\delta d^{p,p'} v_{i,\delta}^{p} \leqslant T_i^{p'} \quad \forall i \in N, p, p' \in P_i \backslash \{P_{iL}\} : \{p \prec p'\} \tag{2.9}$$

$$T_i^p \geqslant Start_i^p \quad \forall i \in N, p \in P_i \tag{2.10}$$

$$\sum_{b\in B_p} T_i^{p,b} + \sum_{b\in B_p} h_i^{p,b}\Big(\sum_{j\in N\cup\{d(p,b)\}} y_{i,j}^{p,b}\Big) - EFT_i^p \leqslant \Delta EFT_i^p \ \forall i \in N, p \in P_i \tag{2.11}$$

$$\sum_{b\in B_p} T_i^{p,b} \geqslant T_i^p \quad \forall i \in N, p \in P_i \tag{2.12}$$

$$\Big(\sum_{j\in N\cup\{d(p,b)\}} y_{i,j}^{p,b} + \sum_{j\in N\cup(o(p,b))} y_{j,i}^{p,b}\Big) M2_i^{p,b} \geqslant T_i^{p,b} \ \forall i \in N, p \in P_i, b \in B_p \tag{2.13}$$

$$T_{o(p,b)}^{p,b} \geqslant s^{p,b} \quad \forall p \in P, b \in B_p \tag{2.14}$$

$$T_{d(p,b)}^{p,b} \leqslant e^{p,b} \quad \forall p \in P, b \in B_p \tag{2.15}$$

$$\sum_{\delta\in S} v_{i,\delta}^{p} = 1 \quad \forall i \in N, p \in P_i \backslash \{P_{iL}\} \tag{2.16}$$

$$y_{i,j}^{p,b} \in \{0,1\} \quad \forall (i,j) \in A^{p,b}, p \in P, b \in B_p \tag{2.17}$$

$$v_{i,\delta}^{p} \in \{0,1\} \quad \forall i \in N, p \in P_i, \delta \in S \tag{2.18}$$

$$\Delta EFT_i^p, T_i^p \in \mathbb{Z}^+ \quad \forall i \in N, p \in P_i \tag{2.19}$$

$$T_{o(p,b)}^{p,b}, T_{d(p,b)}^{p,b} \in \mathbb{Z}^+ \quad \forall p \in P, b \in B_p \tag{2.20}$$

$$T_i^{p,b} \in \mathbb{Z}^+ \quad \forall i \in N, p \in P_i, b \in B_p \tag{2.21}$$

The objective function (2.3) minimizes the cost, both for the terminal operators and the liner shipping company. It consists of the four cost elements described in Section 2.3.2, namely, the cost of waiting at the port, the vessels' handling cost, the cost of delays, and the total cost of the fuel consumed when sailing between

ports. The waiting time is computed as the positive difference between the berthing time and the arrival time whereas the delay is computed as the positive difference between the actual and expected berthing finish time. Constraints (2.4) ensure that each ship berths at only one berth at each port in its route. Constraints (2.5) and (2.6) denote that at each berth and each port, only one arc leaves the origin and one arrives at the destination respectively. The flow conservation for all arcs at each berth and each port is ensured by constraint (2.7). Constraints (2.8) guarantee that if ship j is berthing right after ship i, it waits until the handling is completed. The big-M values for these constraints can be tightened to the time when the berth closes ($e^{p,b}$). Constraints (2.9) ensure for each ship that the activities at the next port in the route do not commence before the ship arrives to the port. The left-hand side of the constraint computes the arrival time to the next port travelling at a chosen speed. The start of activities for each ship at each port must also start after the minimum allowed time ($Start_i^p$) as indicated in constraints (2.10). This also ensures that a ship cannot start berthing if it arrives too early. Both constraints (2.9) and (2.10) set a lower bound (LB) for the variable T_i^p. Constraints (2.11) compute and set the delay (ΔEFT_i^p) for each ship at each port. Constraints (2.12) ensure that the berthing time of each ship at each port starts after the activities for that ship are open at the port. The values of the berthing time variables for the not chosen berths are set to 0 by constraints (2.13). Constraints (2.14) and (2.15) ensure that all berthing periods occur within the time window of each berth. Constraints (2.16) ensure that exactly one speed is selected to travel between each pair of consecutive ports (leg) in the route. The domains for all the decision variables are defined in (2.17)-(2.21). We notice that a formulation where the time-based variables are defined as non-negative real numbers (i.e., $\mathbb{R}^+$) is also valid. However, we maintain the integer property of the variables for a fair comparison with the presented methods and the formulation presented in Venturini et al. (2017).

This formulation contains a few modifications to the original model presented in Venturini et al. (2017) (referred to as *original* model). In the original model a set of additional variables for the arrival of a ship to a port is stated. These variables have been omitted in this formulation since the arrival time of a ship to the next port in the route is directly dependent on the departure time from the previous port and the sailing speed between ports. This calculation is given by the left-hand side of constraints (2.9), which then can be used to replace arrival time variables (e.g., in the objective function). The delay calculation constraints (2.11) use the berthing time ($T_i^{p,b}$) instead of the port opening time for the ship (T_i^p). The big-M value of constraints (2.8) is set to the closing time of the berth ($e^{p,b}$) instead of $e^{p,b} - min_{c\in(i,j)}\{Start_c^p\}$.

Venturini et al. (2017) enhance the original formulation by adding multiple sets

of valid inequalities. These enhancements have also been implemented for the computational comparison. The reader is referred to the original publication for additional details.

2.3.4 Network flow formulation

The MPBAP can also be modeled as a network flow problem using a graph representation where each node represents a feasible berthing time at each port and berth and arcs enable the different combinations of berthing times along the route. This setup allows us to obtain a feasible voyage for a given ship by choosing a path along the ports in the graph. Figure 2.4 shows an illustrative example of such a path. It consists of three ports with either one or two berthing positions in each of them.

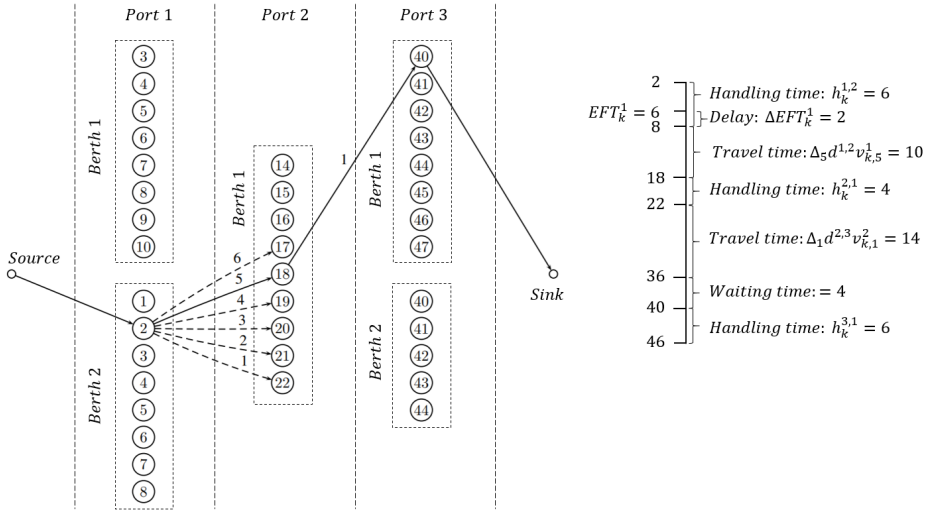


Figure 2.4: Example voyage for ship k and corresponding timeline. The number in the nodes indicate the berthing time and the number on the arcs denotes the speed level chosen. Alternative sailing options are denoted with dashed arcs. The rest of arcs in the graph are not displayed for simplicity.

Let $G = (O, A)$ be a directed and acyclic graph formed by the sets of nodes O and arcs A. Additionaly, we define the subset of arcs $A^k \subseteq A$ which denote the arcs available for a given ship $k \in N$. Within the node set, we denote $o, d \in O$ as artificial source and sink nodes respectively. Let $\delta_k^+(u)$ be the set of nodes that can be reached by following a single outgoing arc $a \in A^k$ from node $u \in O$ for ship $k \in N$. Likewise, let $\delta_k^-(u)$ be the set of nodes that can be reached

by following a single incoming arc $a \in A^k$ from node $u \in O$ for ship $k \in N$. Additionally, $\theta(u)$ denote the berthing time related to node $u \in O \backslash \{o, d\}$ and let $V(p, b) \subseteq O$ be the set of nodes corresponding to port $p \in P$ and berth $b \in B_p$. We use the notation $[x; y]$ to define an interval between x and y where y is included and $[x; y)$ where y is not. For each ship $n \in N$ port $p \in P$ berth $b \in B_p$ and operating time instant $t \in [s^{p,b}; e^{p,b})$, we define the set $C(n, p, b, t) \subseteq V(p, b)$ that denote the graph nodes for ship n whose berthing periods cover time t (i.e., nodes that are *in conflict* with any ship berthing at time t). This basically corresponds to the nodes of the previous $h_n^{p,b} - 1$ time instants and including the node related to time t. An example is depicted in Figure 2.5 and the expression can be stated as follows:

$$C(n, p, b, t) := \left\{ v \in V(p, b) \middle| \theta(v) \in \left[\max\left(t - h_n^{p,b} + 1, s^{p,b}\right); \min\left(t, e^{p,b}\right) \right] \right\}$$

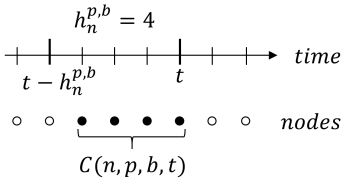


Figure 2.5: An example of the set $C(n, p, b, t)$ where the nodes depicted belong to $V(p, b)$ and refer to the time instant directly above. $h_n^{p,b}$ denotes the handling time for ship n.

Finally, let $x_{i,j}^k$ be a binary variable deciding if arc $(i, j) \in A^k$ is selected for ship $k \in N$ and let $c_{i,j}$ be the weight associated to the same arc.

$$\min \sum_{k \in N} \sum_{(i,j) \in A^k} c_{i,j} x_{i,j}^k \tag{2.22}$$

$$\sum_{j \in \delta_k^+(o)} x_{o,j}^k = 1 \quad \forall k \in N \tag{2.23}$$

$$\sum_{i \in \delta_k^-(d)} x_{i,d}^k = 1 \quad \forall k \in N \tag{2.24}$$

$$\sum_{i \in \delta_k^-(j)} x_{i,j}^k - \sum_{i \in \delta_k^+(j)} x_{j,i}^k = 0 \quad \forall j \in O \backslash \{o, d\}, k \in N \tag{2.25}$$

$$\sum_{k \in N} \sum_{i \in C(k,p,b,t)} \sum_{j \in \delta_k^+(i)} x_{i,j}^k \leq 1 \quad \forall p \in P, b \in B_p, t \in [s^{p,b}; e^{p,b}) \tag{2.26}$$

$$x_{i,j}^k \in \{0, 1\} \quad \forall (i, j) \in A, k \in N \tag{2.27}$$

The objective remains the same, and in this case the objective function (2.22) minimizes the cost of the selected arcs. Constraints (2.23) and (2.24) ensure that, for each ship, only one arc leaves from the source node and arrives to the sink node respectively. Constraints (2.25) enforce flow conservation ensuring that for each node, except the source and sink ones, there are as many incoming as outgoing arcs. Constraints (2.26) avoid overlapping of berthing periods in the same position by at most allowing one ship to be berthing at each time instant. Finally, constraints (2.27) define the binary property of the variable.

2.3.5 Generalized set partitioning problem formulation

It is noted that all constraints of the network flow formulation (2.22)-(2.27) except constraint (2.26) are independent between ships. Exploiting the structure of the formulation, we can apply Dantzig-Wolfe decomposition (Dantzig and Wolfe, 1960) and transform it into a generalized set partitioning problem (GSPP) formulation where constraint (2.26) is handled in the master problem and each variable (i.e., column) refers to a whole feasible schedule of a ship along its route. According to Jans (2010), the pure binary nature of the variables of the network flow formulation allows us to impose binary conditions on the variables of the new master problem.

The set of all columns is comprised in Ω and the decision variable λ_j is set to 1 if column $j \in \Omega$ is chosen as part of the solution and 0 otherwise. We denote c_j as the cost related to column $j \in \Omega$. In order to replicate the objective of the MIP formulation, this cost consists of the idleness, handling cost, delay and bunker consumption cost of the ship denoted by the column. Let A_j^i be a parameter that is equal to 1 if column $j \in \Omega$ corresponds to ship $i \in N$ and 0 otherwise. Likewise, let $Q_j^{p,b,t}$ be a parameter that is equal to 1 if the ship of column $j \in \Omega$ is occupying berth $b \in B_p$ at time instant $t \in [s^{p,b}; e^{p,b})$ at port $p \in P$ and 0 otherwise.

$$\min \sum_{j \in \Omega} c_j \lambda_j \tag{2.28}$$

$$\sum_{j \in \Omega} A_j^i \lambda_j = 1 \quad \forall i \in N \tag{2.29}$$

$$\sum_{j \in \Omega} Q_j^{p,b,t} \lambda_j \leqslant 1 \quad \forall p \in P, b \in B_p, t \in [s^{p,b}; e^{p,b}) \tag{2.30}$$

$$\lambda_j \in \{0,1\} \quad \forall j \in \Omega \tag{2.31}$$

The objective function (2.28) minimizes the cost c_j of the columns. Constraints (2.29) ensure that one column is selected for each ship. Constraints (2.30) guarantee that, at each time instant, there is at most one ship berthing at each berth of a port. Finally, constraints (2.31) set the binary property of the decision variables.

2.4 Solution method

To solve (2.28)-(2.31), we propose a solution method based on a column generation procedure that, combined with branching, additional valid inequalities and symmetry breaking methods, results in a *branch-and-cut-and-price* algorithm.

2.4.1 Delayed column generation

A common way of solving the GSPP formulation is by adding all the columns in advance. A successful example of this approach for the BAP can be found in Buhrkal et al. (2011). For the BAP instances presented, the amount of columns is manageable and can be easily pre-processed. However, in the MPBAP, the amount of columns increase exponentially with the multiple sailing speeds and ports for a ship. This makes the pre-processing intractable even for a few ports. Therefore, more dynamic strategies for handling the columns need to be explored. One efficient procedure is the so-called *delayed column generation*. This procedure relies on the premise that most of the variables will not be part of the optimal solution and, therefore, have a value of zero. Then, the focus is only on generating columns that have the potential to improve the objective value. This is done by relaxing and splitting the main problem into two, the master and subproblem. The restricted master problem (RMP) is the linear relaxation of the original formulation containing only a subset of the variables. The subproblem (or pricing problem) is used to identify the new variables. In our case, the relaxed version of the GSPP becomes the RMP and we define N independent subproblems, one per ship. The subproblem is defined as a shortest path problem in the network defined in Section 2.3.4 which can be solved in polynomial time. Since the graph is directed and acyclic (DAG), it can be solved by a DAG shortest path algorithm (see Cormen et al. (1996) or Magnanti et al. (1993)). The pricing problem aims at minimizing the reduced cost of a given path. At each iteration, after solving the RMP, the dual values of the RMP constraints are used to solve the pricing problems. We denote α_k to the dual variable for ship $k \in N$ associated to constraint (2.29). Likewise, we denote $\mu_{p,b,t}$ to the dual variable for port $p \in P$, berth $b \in B_p$ and time $t \in [s^{p,b}; e^{p,b})$ associated to constraint (2.30). Let $\bar{\alpha}_k, \bar{\mu}_{p,b,t}$ be the dual solution values for the RMP and let Λ_j be a sequence of *(port,berth,time)* elements. Each of these elements refers to the port, berth and time of a graph node visited by column $j \in \Omega$. The reduced cost $\hat{c}_j$ for a specific path j for ship $k \in N$ is computed as follows:

$$\hat{c}_j = c_j - \Big(\sum_{(p,b,t)\in\Lambda_j} \sum_{t'\in[t;t+h_k^{p,b})} \bar{\mu}_{p,b,t'} \Big) - \bar{\alpha}_k$$

Finally, for each pricing problem, we add the path with the lowest reduced cost to the RMP only if $\hat{c}_j$ is negative (i.e, $\hat{c}_j < 0$).

In fact, when the pricing problem is a pure shortest path problem, the LP bound arising from solving the GSPP with column generation and solving the LP relaxation of the network flow problem is the same. This indicates that the Dantzig-Wolfe decomposition does not provide any gain bound-wise. On the other hand, in cases of very dense networks with significantly more arcs than nodes as in our case, solving the GSPP with column generation is expected to be faster (e.g., see Brouer et al. (2011)).

2.4.2 Branching

Since the decision variables of the RMP are linear, the solution at the root node is often fractional and branching methods are required in order to achieve integrality. A major aspect of the branching procedure is selecting a branching candidate, whose branch children improve the lower bound the most. The most common branching methods consider branching on a specific node or arc from the graph. These strategies can be effective in some cases but do not necessarily apply to our problem. For instance, when branching on a graph node, one child will enforce the graph node to be used in the subsequent branch-and-bound (B&B) tree while the other child will forbid it. Considering the large amount of nodes for most instances in this problem, we can clearly see that the effect can be significant for the first child but rather minimal for the second. This often results in a highly unbalanced B&B tree to explore. In this study, we present a different branching strategy for the problem at hand that aims to be more effective than branching on a single graph node.

The proposed branching strategy states that, given a fractional solution, we compute, for each ship n and port p, the average berthing time t and the variance of these times among all solution columns. As an example, consider a fractional solution containing two columns for ship 1. At port 1, these columns correspond to ship 1 berthing at time 4 and 6 respectively. Then, for ship 1 and port 1, the average berthing time is 5 whereas the sample standard deviation is $\sqrt{2}$. We define this average berthing time and variance as a *candidate* which results in a total of $|N| \cdot |P|$ candidates. We then select the candidate whose variance of berthing times is higher. The procedure is described in Algorithm 2.1. Each of the child branches will enforce ship n to berth before or after time t respectively at port p. It should be noticed that a fractional solution where ships berth at the same time but at different berthing positions can exist. In this case, we can obtain candidates with no variance resulting in an impractical branching. If that happens, the criterion is changed to branching on berthing positions instead of on berthing times, following the same procedure. In practice, this scenario is highly unlikely to happen and we have not experienced it in any experiments hitherto. As a result, the proposed strategy opts for branching on a set of graph nodes instead of on a single one.

Finally, the B&B tree is explored following a *best first* policy. This policy prioritizes the queue of unexplored nodes according to their bound. Thus, the next node to be explored is always the one with the *best* (i.e., lowest) lower bound.

Algorithm 2.1 Branching candidate selection

```
1: procedure SELECTCANDIDATE(sol)(current solution)
2:     [λ] ← sol                          ▷ classify solution columns (λ) by ship
3:     Cand* = ∅                          ▷ initialize best candidate
4:     σ* = 0          ▷ initialize standard deviation of candidate's berthing times
5:     for all ships and ports do
6:         [times] ← λ(ship, port)  ▷ set of solution berthing times at port for ship
7:         time ← avg([times])                ▷ get average berth time
8:         if σ([times]) > σ* then            ▷ compare the standard deviation
                                                  with the current best
9:             σ* ← σ([times])
10:            Cand* ← time, port, ship       ▷ update best candidate so far
11:        end if
12:    end for
13:    return Cand*
14: end procedure
```

2.4.3 Valid inequalities

In order to improve the lower bound, we propose a set of valid inequalities that can be added to the problem by separation.

Figure 2.6 shows a small LP solution to a trivial problem with two ships (i.e., continuous and dashed lines), one port and one berth where an example of a violated valid inequality can be found. We define u, v as the two nodes corresponding to ship A (berthing at times 1 and 3) and let w be the node of ship B berthing at time 2. We observe that the arc from node w is in conflict with the arcs from both nodes u and v due to overlapping berthing periods. In other words, the berthing period of ship B at node w covers, at least partially, both berthing periods of ship A at nodes u and v. The arcs from u, v are also in conflict with each other as they belong to the same ship. As a result, we notice that, at most, one outgoing arc can be chosen out of the ones from these three nodes. Since the solution values of the outgoing arcs sum to 1.5, this valid inequality would cut the example LP solution. We aim at generalizing the definition of such a valid inequality and introduce the following proposition:

Proposition 2.1 *Given two time instants $t_1, t_2 \in [s^{p,b}; e^{p,b})$ where $t_1 < t_2$ and a port $p \in P$, berth $b \in B_p$ and ship $n \in N$, the following is a valid*

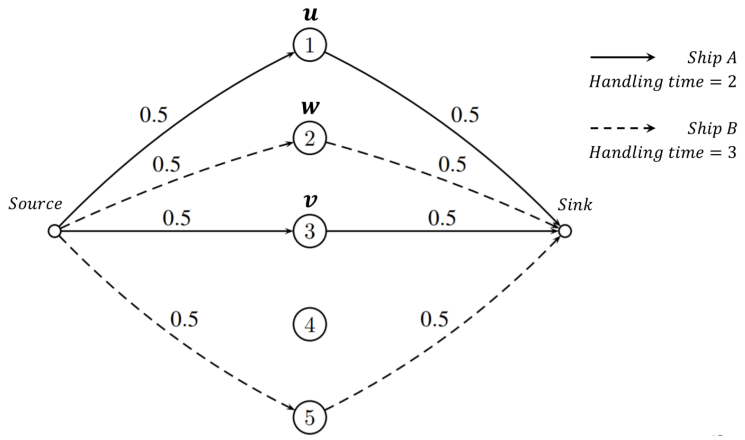


Figure 2.6: Example LP solution of a problem with two ships, one port and one berth. The nodes represent berthing times and the numbers on the arcs denote the solution value of the arc variable $x^k_{i,j}$.

inequality:

$$\sum_{u \in \bigcup_{t \in [t_1;t_2]} C(n,p,b,t)} \sum_{w \in \delta^+_n(u)} x^n_{u,w} + \sum_{m \in N \setminus \{n\}} \sum_{u \in C(m,p,b,t_1) \cap C(m,p,b,t_2)} \sum_{w \in \delta^+_m(u)} x^m_{u,w} \leqslant 1$$

Proof. The set $C(m,p,b,t)$ used in constraint (2.26) defines the set of nodes for ship m that are in conflict with time t (see Section 2.3.4). Based on this definition, the intersection set $C(m,p,b,t_1) \cap C(m,p,b,t_2)$ directly defines the set of nodes for ship m that are in conflict with both time instants t_1 and t_2. Constraint (2.26) indicates that at most one arc can be chosen out of the nodes from the sets $C(m,p,b,t)$ of all ships $m \in N$ and, therefore, the same applies to the intersection set $C(m,p,b,t_1) \cap C(m,p,b,t_2)$. By considering the intersection set $C(m,p,b,t_1) \cap C(m,p,b,t_2)$ for all ships except one $m \in N \setminus \{n\}$, the berthing period for ship n is only required to be in conflict with either t_1 or t_2 and can be defined as the union of $C(n,p,b,t_1) \cup C(n,p,b,t_2)$. Considering these node sets, we can define the following valid inequality:

$$\sum_{u \in C(n,p,b,t_1) \cup C(n,p,b,t_2)} \sum_{w \in \delta^+_n(u)} x^n_{u,w} + \sum_{m \in N \setminus \{n\}} \sum_{u \in C(m,p,b,t_1) \cap C(m,p,b,t_2)} \sum_{w \in \delta^+_m(u)} x^m_{u,w} \leqslant 1$$

$$\forall p \in P, b \in B_p, n \in N, t_1, t_2 \in [s^{p,b}; e^{p,b}), t_1 < t_2$$

Based on the assumption that a berthing period cannot be discontinued, the intersection set $C(m,p,b,t_1) \cap C(m,p,b,t_2)$ for any ship is not only in conflict with times t_1 and t_2 but with all the time instants in the period $[t_1; t_2]$. Therefore

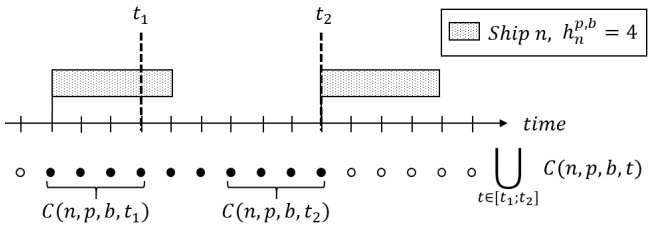


Figure 2.7: In a valid inequality for ship n, port p, berth b and time instants t_1, t_2, the filled nodes indicate the interval for ship n with handling time $h_n^{p,b}$. The rectangles indicate the berthing period of ship n at the earliest and latest possible berthing times in the interval.

the interval for ship n can be expanded to the union of $C(n,p,b,t)$ sets for all time instants $t \in [t_1;t_2]$. An example of this set is shown in Figure 2.7 and the resulting valid inequality can be formulated as follows:

$$\sum_{u \in \bigcup_{t \in [t_1;t_2]} C(n,p,b,t)} \sum_{w \in \delta_n^+(u)} x_{u,w}^n + \sum_{m \in N \setminus \{n\}} \sum_{u \in C(m,p,b,t_1) \cap C(m,p,b,t_2)} \sum_{w \in \delta_m^+(u)} x_{u,w}^m \leqslant 1$$
$$\forall p \in P, b \in B_p, n \in N, t_1, t_2 \in [s^{p,b}; e^{p,b}), t_1 < t_2 \quad (2.32)$$

□

Returning to the example in Figure 2.6, the mentioned cut would be included in the proposed valid inequality (2.32) for $n = A$, $t_1 = 2$ and $t_2 = 4$ where node w would correspond to a node from the intersection sets $C(B,p,b,2) \cap C(B,p,b,4)$ and nodes u, v for ship A would correspond to berthing times covering t_1 and t_2 respectively and therefore belonging to the set $\bigcup_{t \in [2;4]} C(A,p,b,t)$.

We note that the inequality only is interesting when $C(m,p,b,t_1) \cap C(m,p,b,t_2) \neq \emptyset$. The size of the intersection set is dependent on the time instants t_1, t_2 used and we observe that this size increases when the t_1, t_2 are closer together in time.

These valid inequalities (2.32) are added by separation after the column generation procedure concludes. Exploring the entire set of valid inequalities can be computationally intensive. Therefore, only valid inequalities based on berthing times from the LP solution are checked since the arcs from the related nodes are guaranteed to contain non-zero values and the resulting inequalities have a higher probability of being violated by the LP solution. Given an LP solution, let t_1^* and t_2^* be two berthing times for ship n at berth b of port p where $t_1^* \leq t_2^*$. Let t_3^* be a berthing time for another ship m at the same berth b of port p whose

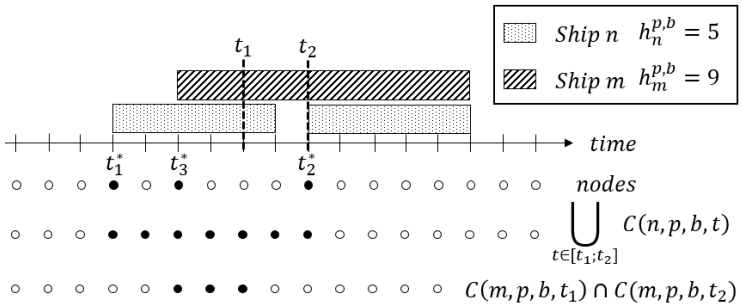


Figure 2.8: Example of times t_1, t_2 definition based on solution times t_1^*, t_2^* for ship n and t_3^* for ship m. The bottom last two rows of filled nodes define the node interval for ship n and m respectively.

berthing period is both in conflict with t_1^* and t_2^* for ship n. The conditions that t_3^* needs to satisfy to be in conflict with t_1^* and t_2^* are given by the following inequalities:

$$t_1^* + h_n^{p,b} > t_3^*$$
$$t_2^* < t_3^* + h_m^{p,b}$$

Based on these times, we can calculate time instants t_1, t_2 for a valid inequality that includes t_1^*, t_2^* for ship n and t_3^* for ship m as follows:

$$t_1 = t_1^* + h_n^{p,b} - 1, \quad t_2 = t_2^*$$

An example of this calculation is shown in Figure 2.8. It can be noticed that the interval for ship n starts at time t_1^* and ends at time t_2^*. If we add such a violated cut to the RMP, we risk finding a very similar solution in the next iteration where columns are shifted, for example, one time instant before t_1^* or after t_2^*. In order to avoid that, we aim at defining time instants t_1, t_2, so that the resulting intervals do not only cover solution nodes but also a number of neighboring nodes related to time instants immediately before and after the solution time. We aim at expanding the interval between t_1^* and t_2^* as well as the one around t_3^*. Based on the inequalities aforementioned to ensure that t_1^*, t_2^* and t_3^* relate to conflicting periods, we introduce the slack variables Δ^X and Δ^Y that would indicate how much we can modify the node intervals.

$$t_1^* + h_n^{p,b} > t_3^* + \Delta^X$$
$$t_2^* + \Delta^Y < t_3^* + h_m^{p,b}$$

Both slack values are distributed equally between both intervals, which leads us to the following calculation of t_1 and t_2:

$$t_1 = t_1^* - \frac{\Delta^X}{2} + h_n^{p,b} - 1, \quad t_2 = t_2^* + \frac{\Delta^Y}{2}$$

Due to the discretization of the time horizon, if $\frac{\Delta^X}{2}$ or $\frac{\Delta^Y}{2}$ is fractional, they are rounded-up in the calculation of t_1 and t_2. Also, in the case that there is limited

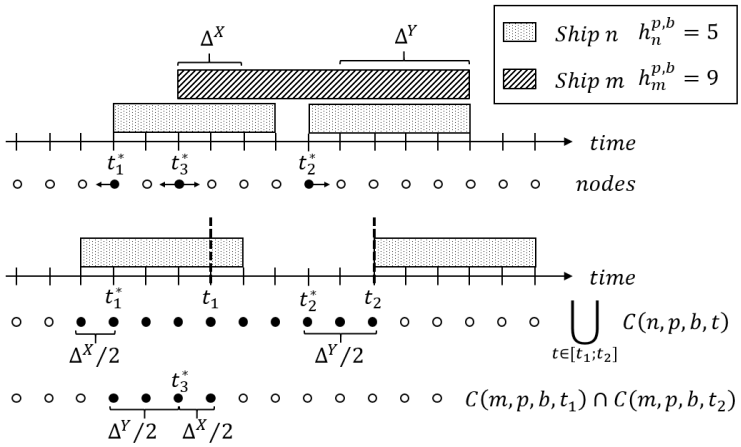


Figure 2.9: Example of times t_1, t_2 selection based on solution times t_1^*, t_2^* for ship n and t_3^* for ship m. The upper illustration depicts the LP solution for the three berthing times selected and the available slack and direction of expansion for the desired node intervals. The lower illustration depicts the resulting times t_1, t_2 for the valid inequality and the respective node intervals for ships n and m.

room for expansion in one of the intervals (e.g., operational time windows), the remaining slack is added to the other interval.

Figure 2.9 shows an example of the calculation of times t_1 and t_2 based on solution times t_1^*, t_2^* and t_3^* and slack variables Δ^X, Δ^Y.

In order to ensure $t_1 < t_2$, by substituting the above expressions, the criterion that t_1^*, t_2^*, t_3^* need to fulfill in order to result in a valid inequality can be defined as follows:

$$t_1^* - \frac{\Delta^X}{2} + h_n^{p,b} - 1 < t_2^* + \frac{\Delta^Y}{2}$$

Not satisfying this inequality leads to a cut that, at best, is equal to constraint (2.30) which is already present in the RMP.

The entire cut separation process is summarized in Algorithm 2.2. The procedure requires the RMP model and an LP solution as input. From the solution, both the λ^* solution values and the berthing times of the solution columns are extracted and classified by ship, port and berthing position. The cuts are checked by enumerating combinations of solution times t_1^*, t_2^* and t_3^*. Only solution times whose berthing periods are in conflict are considered. This is the case if the berthing period of ship m at time t_3^* overlaps both berthing periods of ship n at times t_1^* and t_2^* ($inConflict(t_1^*, t_2^*, t_3^*)$ in Algorithm 2.2). Then, the solution

times are used to compute time instants t_1, t_2 distributing the slack available as aforementioned in this Section ($t_1, t_2 \leftarrow calcInterval(t_1^*, t_2^*, t_3^*)$ in Algorithm 2.2). To check and add the violated cuts to the RMP, equation (2.32) needs to be translated to the problem variables. The $x_{i,j}^k$ variables can be defined using λ_p variables as follows:

$$x_{i,j}^k = \sum_{p \in \Omega} q_{i,j,p}^k \lambda_p$$

where parameter $q_{i,j,p}^k$ is 1 if graph arc $(i,j) \in A^k$ for ship k is used by column $p \in \Omega$ and 0 otherwise. Applying this equality to equation (2.32), we obtain the following version of the equation:

$$\sum_{u \in \bigcup_{t \in [t_1;t_2]} C(n,p,b,t)} \sum_{w \in \delta_n^+(u)} \sum_{j \in \Omega} q_{u,w,j}^n \lambda_j + \sum_{m \in N \setminus \{n\}} \sum_{u \in C(m,p,b,t_1) \cap C(m,p,b,t_2)} \sum_{w \in \delta_m^+(u)} \sum_{j \in \Omega} q_{u,w,j}^m \lambda_j \leq 1 \tag{2.33}$$

For each cut inspected, the left-hand side of constraint (2.33) is computed and the valid inequality is added to the RMP if it is violated.

These valid inequalities are relatively easy to handle in the reduced cost computation. For each valid inequality, its corresponding dual value needs to be subtracted in each of the nodes considered for each ship in the constraint. As an example, given a valid inequality for times t_1, t_2 where $t_1 < t_2$, port p, berth b and ship n, its dual value needs to be subtracted in nodes $\bigcup_{t \in [t_1;t_2]} C(n,p,b,t)$ for ship n and in nodes $C(m,p,b,t_1) \cap C(m,p,b,t_2)$ for ship m where $m \neq n$. A more mathematical definition of the updated reduced cost computation is given in Appendix 2.A.1.

2.4.4 Symmetry breaking

In some instances, at each port, some of the berthing positions are identical in terms of their availability time window and the handling times for all ships. Identical berths may lead to many equivalent solutions, which may increase the solving time of the model. Therefore, we propose adapting the model so it deals with berth types instead of individual berths in a similar procedure as the one stated in Buhrkal et al. (2011). Let K_p be the set of berth types for port $p \in P$ and β^k be the number of berthing positions of type $k \in K$ in the problem. For each berth type $k \in K_p$ at port $p \in P$, $s^{p,k}$ and $e^{p,k}$ denote its opening and closing time respectively and the parameter $Q_j^{p,k,t}$ is 1 if the ship from column $j \in \Omega$ occupies berth type k at time instant $t \in [s^{p,k}; e^{p,k})$ at port p and 0 otherwise. We can therefore update the set of constraints (2.30) as follows:

$$\sum_{j \in \Omega} Q_j^{p,k,t} \lambda_j \leqslant \beta^k \quad \forall p \in P, k \in K_p, t \in [s^{p,k}; e^{p,k}) \tag{2.34}$$

Algorithm 2.2 Cut separation

procedure CUTSEPARATION(sol, RMP)(current solution and model)
  $times[p, b, n] \leftarrow sol$ ▷ divide solution times by port p, berth b and ship n
  $[\lambda^*] \leftarrow sol$ ▷ obtain solution values for columns
  for all $p \in P, b \in B_p, n \in N$ **do** ▷ cuts are based on a specific berth, port and ship
    for all $t_1^*, t_2^* \in times[p, b, n]$ **do** ▷ loop over pairs of solution times for ship n
      for all $m \in N, m \neq n$ **do**
        for all $t_3^* \in times[p, b, m]$ **do** ▷ select a third time from a different ship
          if inConflict(t_1^*, t_2^*, t_3^*) **then** ▷ check if berthing periods are in conflict
            $t_1, t_2 \leftarrow$ calcInterval(t_1^*, t_2^*, t_3^*) ▷ compute t_1, t_2 for the valid inequality
            $violatedCut \leftarrow$ checkCut$(t_1, t_2, n, p, b, [\lambda^*])$ ▷ compute constraint (2.33)
            if $violatedCut \neq \emptyset$ **then**
              $RMP \leftarrow violatedCut$ ▷ add violated cut to the RMP
            end if
          end if
        end for
      end for
    end for
  end for
  return RMP ▷ return the updated problem
end procedure

This adaptation has an equivalent impact in constraints (2.26) from the network formulation where the right-hand side is also replaced by β^k. The valid inequality (2.33) from Proposition 2.1 can be updated similarly and the proposition and corresponding proof can be found in Appendix 2.A.2. The resulting valid inequality is formulated as follows:

$$\sum_{u \in \bigcup_{t \in [t_1;t_2]} C(n,p,k,t)} \sum_{w \in \delta_n^+(u)} \sum_{j \in \Omega} q_{u,w,j}^n \lambda_j + \sum_{m \in N \setminus \{n\}} \sum_{u \in C(m,p,k,t_1) \cap C(m,p,k,t_2)} \sum_{w \in \delta_m^+(u)} \sum_{j \in \Omega} q_{u,w,j}^m \lambda_j \leq \beta^k$$
$$\forall p \in P, k \in K_p, n \in N, t_1, t_2 \in [s^{p,k}; e^{p,k}), t_1 < t_2 \tag{2.35}$$

The reduced cost computation is also slightly modified where the dual variable $\mu_{p,k,t}$ of the modified constraint now is based on berth type $k \in K_p$ instead of berth $b \in B_p$.

We expect to see an improvement in the computational time as soon as there are two identical berths at a port. Likewise, we expect to see larger symmetry for the instances containing more berthing positions per port.

2.5 Cooperative game theory

The MPBAP is based on a strong collaboration between carriers and port operators and some of them, especially carriers, may be reticent to take part in such a collaboration scheme. In order to convince them that this form of collaboration is beneficial for all of them, we define a cooperative game. The aim is to show that all stakeholders (i.e., carriers and terminals) can potentially benefit from a collaboration by distributing the overall costs efficiently. Our cooperative game is formed by a set of players $\mathcal{P} = \{1, ..., p\}$, which in this case corresponds to both the carriers owning the ships and the terminal operators of the ports visited by the ships. The *characteristic function* $\vartheta(S)$ measures the impact of a coalition of players $\mathcal{S} \subseteq \mathcal{P}$, which in this case is measured by the operational costs. The coalition formed by all players is known as the *grand coalition*. It is normally assumed that the characteristic function satisfies:

$$\vartheta(\emptyset) = 0 \tag{2.36}$$

$$\vartheta(\mathcal{S} \cup \mathcal{T}) \leq \vartheta(\mathcal{S}) + \vartheta(\mathcal{T}) \quad \forall \mathcal{S}, \mathcal{T} \subseteq \mathcal{P}, \quad \mathcal{S} \cap \mathcal{T} = \emptyset \tag{2.37}$$

Equation (2.36) states that an empty coalition has a cost of zero, while equation (2.37), known as *subadditivity*, indicates that the costs of two separate coalitions $\mathcal{S}, \mathcal{T} \subseteq \mathcal{P}$ cannot be lower than when acting together. A solution to a cooperative game (i.e., *imputation*) can be defined as $f = \{f_1, ..., f_p\}$ where f_i corresponds to the cost allocation of player i in coalition $\mathcal{P}$. An imputation should satisfy the following conditions:

$$f_i \leq \vartheta(\{i\}) \quad \forall i \in \mathcal{P} \tag{2.38}$$

$$\sum_{i \in \mathcal{P}} f_i = \vartheta(\mathcal{P}) \tag{2.39}$$

The first condition is based on individual rationality and defines that the cost allocation for a player when being part of the grand coalition cannot be worse than the player's standalone cost. The second condition is based on group rationality and states that all the savings arising from a grand coalition are shared. This is the equivalent of saying that the sum of cost allocations needs to be equal to the total cost of the grand coalition and a solution fulfilling this condition is said to be *efficient*. Furthermore, we consider a solution to be *stable*, if, for every coalition $\mathcal{S} \subseteq \mathcal{P}$, the sum of allocated cost of the players of the coalition is not higher than the cost of the coalition $\sum_{k \in \mathcal{S}} f_k \leq \vartheta(\mathcal{S})$. We define the *core* as the set of solutions that are both *efficient* and *stable*. We see the core solutions as the most attractive and fair for all players. Note, however,

that the core may be empty in some cases. This means that a cost allocation that satisfies both the efficiency and stability properties does not exist. In other words, it means that a subset of the players in the grand coalition could do better by themselves (i.e., by forming a sub-coalition). If the core is empty, the grand coalition is unstable and there is a risk that it breaks apart. In practice, the grand coalition may stay together despite a non-core solution. For instance, it may be that a subset of players are not aware of the higher benefits of a specific sub-coalition or that players choose to stay in the coalition to reap more long-term benefits given future expectations. Next, we describe the two allocation methods we have used in this study.

2.5.1 Shapley value

The Shapley value (Shapley, 1953), refers to the weighted average of each player's marginal contribution to each of the potential coalitions. Let $\Theta^i(\mathcal{S})$ be the marginal contribution of player i to coalition $\mathcal{S}$, which is seen as the difference between the cost of the coalition including player i and the coalition without the player:

$$\Theta^i(\mathcal{S}) = \vartheta(\mathcal{S} \cup \{i\}) - \vartheta(\mathcal{S}) \tag{2.40}$$

Then, the cost allocated to participant i is computed by the following expression:

$$f_i = \sum_{\mathcal{S} \subseteq \mathcal{P} \setminus \{i\}} \frac{|\mathcal{S}|!|\mathcal{P} \setminus (\mathcal{S} \cup \{i\})|!}{|\mathcal{P}|!} \Theta^i(\mathcal{S}) \tag{2.41}$$

where $|\cdot|$ refers to the number of players in the given coalition. Once the characteristic function $\vartheta(\mathcal{S})$ is calculated for all possible coalitions $\mathcal{S}$, it is a simple method to compute as it only requires applying a formula. The Shapley value does not only provide *efficient* solutions, it also contains other valuable properties. The solutions are *symmetric* meaning that if two players contribute equally to the coalitions, they achieve the same savings. *Anonimity* is also ensured, which states that the order or labelling of players does not have an impact on the assignment of savings. This property ensures a unique solution which avoids players to regret their choices and prevents additional negotiation processes. On the other hand, the Shapley value does not ensure the *stability* property, meaning that the solution is not guaranteed to be part of the core.

2.5.2 Equal profit method (EPM)

The goal of the equal profit method (Frisk et al., 2010) is to find the solution in the core that minimizes the maximal difference in relative savings between pairs of players. The relative saving of player i is computed as $\frac{\vartheta(\{i\}) - f_i}{\vartheta(\{i\})}$. The method is formulated as the following linear programming model:

$$\min z \tag{2.42}$$

$$z \geq \frac{f_i}{\vartheta(\{i\})} - \frac{f_j}{\vartheta(\{j\})} \quad \forall i, j \in \mathcal{P} \tag{2.43}$$

$$\sum_{i \in \mathcal{P}} f_i = \vartheta(\mathcal{P}) \tag{2.44}$$

$$\sum_{i \in \mathcal{S}} f_i \leq \vartheta(\mathcal{S}) \quad \forall \mathcal{S} \subseteq \mathcal{P} \tag{2.45}$$

$$f_i \geq 0 \quad \forall i \in \mathcal{P} \tag{2.46}$$

Constraints (2.43) calculate the difference in relative savings between each pair of players and restricts z to the largest of those differences. Note that constraints (2.44) and (2.45) are the ones denoting the stability and efficiency properties which means that the EPM method only allows solutions lying in the core.

2.6 Computational results

This section is divided in two. First, the performance of the proposed method is compared to a commercial solver on the set of instances from Venturini et al. (2017) and an additional generated set of harder instances. The second part covers the results of the cost allocation methods for the cooperative game.

2.6.1 Instance results

Different versions of the algorithm have been tested varying the size of the B&B tree where valid inequalities can be added. We consider (i) a pure *branch-and-price* where cut separation is not performed at all, (ii) a partial *branch-and-cut-and-price* where we only allow valid inequalities to be added in the root node, and (iii) a pure *branch-and-cut-and-price* where cuts can be added in all the explored nodes. The RMP model solved is comprised by equations (2.28),(2.29),(2.34), the linear relaxation of (2.31) and valid inequalities (2.35) that are added by separation. The algorithm includes a running time-limit and, if it is reached and a gap between the lower and upper bounds still exists, the GSPP formulation problem is solved with all the generated columns in the B&B tree. This helps tightening the upper bound but requires the integer problem to be solvable in reasonable time. The running time for solving the GSPP is set to 10% of the algorithm running time. Two algorithm time limits of 5 minutes and 3 hours have been tested with an additional (if required) 30 seconds and 18 minutes respectively for solving the GSPP. The model has been entirely written in *Julia* language (Bezanson et al., 2017), modelled using *JuMP* (Dunning et al., 2017) and using *CPLEX v. 12.9* as the solver, allowing 4 threads. It has been tested in an 2.20 GHz Intel Xeon Processor 2650v4 using 4 cores with 32 GB of memory per core. The MIP formulation from Venturini

et al. (2017) has been run in the same machine and solved with the same solver for a fair comparison. The results are summarized in Tables 2.1, 2.2, 2.3 and 2.4, that contain the performance comparison on the benchmark instances from Venturini et al. (2017) and the generated set of harder instances with both algorithm time limits. An instance is represented indicating the number of ships N, the number of berthing positions per port B, the number of ports P and if the time windows TW are tight T or loose L. As indicated in Venturini et al. (2017) a loose time window is approximately 3 times longer than a tight one. In each instance, all ports have the same amount of berthing positions and all the ships follow the same route and have the same speed profiles but both the MIP and GSPP formulations can account for different amount of berthing positions per port, different ship routes and different ship types. The set S is discretized in 11 speed levels, covering the range 14-19 knots. A very low sulphur fuel oil (VLSFO) is used by the ships which is in accordance with the increasing need of ships to reduce their sulphur emissions. Its price (F_c) is computed as the average global price during the first quarter of 2021 corresponding to 500 \$/ton (Ship & Bunker, 2021). Regarding the cost of the different operational aspects at port, the current literature does not provide a consensus on the costs of waiting, handling and delay time. Moreover, this may fluctuate significantly between ports and in many cases they are not made available to the public due to contractual agreements. Meisel and Bierwirth (2009) proposes a delay cost of 1000-3000 \$/hour depending on the ship size and a service cost per quay crane hour of 100 \$. They also consider a speeding-up cost to berth at an earlier time of 1000-3000 \$/hour which can resemble the waiting time cost considered in this study. Venturini et al. (2017) set the terminal handling cost weight to 200 \$/hour and charge an additional 300 \$/hour when there is a delay. They set the cost of waiting one hour at anchorage to 200 \$/hour. For the sake of a fair comparison, we use the same costs as Venturini et al. (2017) which correspond to $H_c = 200$, $D_c = 300$ and $I_c = 200$. LB denotes the best lower bound found whereas Z indicates the best integer solution (i.e., upper bound). The optimality gap is stated under the column *Gap* and it is calculated using the optimal solution, or in the case that this is unknown, the best known solution. The computational time in seconds is given under column T.

Table 2.1: Computational results on instances from Venturini et al. (2017) with a total time limit of 5 minutes and 30 seconds. The MIP formulation is compared to the variants of the presented branch-and-cut-and-price method. "-" means that no integer solution has been found within the time limit. "*" means the time limit has been reached. The best running time is highlighted in bold for instances solved to optimality and the best optimality gap for the rest of instances.

Instance	**MIP formulation**				**Branch & Price**				**Branch & Cut (root node) & Price**				**Branch & Cut & Price**			
N-B-P-TW	**LB**	**Z**	**Gap (%)**	**T (s)**	**LB**	**Z**	**Gap (%)**	**T (s)**	**LB**	**Z**	**Gap (%)**	**T (s)**	**LB**	**Z**	**Gap (%)**	**T (s)**
4-3-3-L	296,600	296,600	0.00	**0.1**	296,600	296,600	0.00	0.5	296,600	296,600	0.00	0.2	296,600	296,600	0.00	0.2
5-3-3-L	394,300	394,300	0.00	**0.4**	394,300	394,300	0.00	3.2	394,300	394,300	0.00	7.2	394,300	394,300	0.00	7.2
6-3-3-L	421,720	421,720	0.00	2.3	421,720	421,720	0.00	0.8	421,720	421,720	0.00	**0.5**	421,720	421,720	0.00	**0.5**
6-3-4-L	647,480	647,480	0.00	89.2	647,480	647,480	0.00	**39.0**	647,480	647,480	0.00	111.7	647,480	647,480	0.00	103.7
10-4-4-L	1,014,437	1,060,900	3.80	*	1,053,030	1,054,700	0.14	*	1,053,092	1,055,300	0.13	*	1,053,295	1,055,000	**0.11**	*
10-4-3-L	689,858	700,000	1.18	*	698,100	698,100	0.00	193.6	698,100	698,100	0.00	**72.7**	698,100	698,100	0.00	115.6
4-4-4-L	405,120	405,120	0.00	**0.3**	405,120	405,120	0.00	0.6	405,120	405,120	0.00	0.6	405,120	405,120	0.00	0.6
5-4-4-L	500,600	500,600	0.00	**0.4**	500,600	500,600	0.00	0.9	500,600	500,600	0.00	0.8	500,600	500,600	0.00	0.8
6-4-4-L	599,980	599,980	0.00	**1.2**	599,980	599,980	0.00	7.6	599,980	599,980	0.00	5.1	599,980	599,980	0.00	5.3
12-5-3-L	811,139	840,640	2.32	*	830,440	830,440	0.00	136.0	830,440	830,440	0.00	**112.1**	830,440	830,440	0.00	120.4
10-6-3-L	680,600	680,600	0.00	219.4	680,600	680,600	0.00	9.7	680,600	680,600	0.00	**5.1**	680,600	680,600	0.00	5.8
11-6-3-L	740,430	749,620	0.78	*	746,220	746,220	0.00	22.5	746,220	746,220	0.00	**12.7**	746,220	746,220	0.00	14.3
12-6-3-L	805,930	810,740	0.48	*	809,840	809,840	0.00	112.6	809,840	809,840	0.00	79.0	809,840	809,840	0.00	**72.5**
10-5-4-L	1,006,635	1,031,100	2.11	*	1,027,592	1,028,320	0.07	*	1,028,194	1,028,320	**0.01**	*	1,027,233	1,028,320	0.11	*
15-10-3-L	1,006,000	1,006,200	0.02	*	1,006,200	1,006,200	0.00	46.4	1,006,200	1,006,200	0.00	27.8	1,006,200	1,006,200	0.00	**25.0**
15-12-3-L	1,001,200	1,002,800	0.16	*	1,002,800	1,002,800	0.00	**1.5**	1,002,800	1,002,800	0.00	**1.5**	1,002,800	1,002,800	0.00	**1.5**
15-10-4-L	1,459,400	1,459,600	0.01	*	1,459,600	1,459,600	0.00	23.8	1,459,600	1,459,600	0.00	40.2	1,459,600	1,459,600	0.00	**13.5**
20-10-3-L	1,341,640	-	0.23	*	1,344,446	1,344,800	0.03	*	1,344,450	1,344,800	0.03	*	1,344,467	1,344,800	**0.02**	*
20-12-3-L	1,331,640	1,343,000	0.36	*	1,336,400	1,336,400	0.00	**2.1**	1,336,400	1,336,400	0.00	2.2	1,336,400	1,336,400	0.00	2.2
4-3-3-T	318,440	318,440	0.00	0.3	318,440	318,440	0.00	0.7	318,440	318,440	0.00	**0.2**	318,440	318,440	0.00	**0.2**
5-3-3-T	405,240	405,240	0.00	**0.6**	405,240	405,240	0.00	1.5	405,240	405,240	0.00	1.3	405,240	405,240	0.00	1.1
6-3-3-T	510,920	510,920	0.00	4.0	510,920	510,920	0.00	1.9	510,920	510,920	0.00	0.9	510,920	510,920	0.00	**0.8**
6-3-4-T	993,460	993,460	0.00	3.5	993,460	993,460	0.00	**1.2**	993,460	993,460	0.00	1.3	993,460	993,460	0.00	**1.2**
10-4-4-T	1,574,771	1,676,990	5.17	*	1,660,640	1,660,640	0.00	101.3	1,660,640	1,660,640	0.00	63.9	1,660,640	1,660,640	0.00	**61.0**
10-4-3-T	973,445	1,023,890	4.77	*	1,022,200	1,022,200	0.00	12.3	1,022,200	1,022,200	0.00	**6.6**	1,022,200	1,022,200	0.00	7.9
4-4-4-T	442,600	442,600	0.00	0.9	442,600	442,600	0.00	1.1	442,600	442,600	0.00	0.7	442,600	442,600	0.00	**0.6**
5-4-4-T	576,010	576,010	0.00	**4.1**	576,010	576,010	0.00	10.3	576,010	576,010	0.00	6.0	576,010	576,010	0.00	6.6
6-4-4-T	653,560	653,560	0.00	11.5	653,560	653,560	0.00	23.5	653,560	653,560	0.00	**8.8**	653,560	653,560	0.00	10.4
12-5-3-T	811,240	835,740	2.31	*	830,440	830,440	0.00	128.0	830,440	830,440	0.00	**68.3**	830,440	830,440	0.00	96.1
12-6-3-T	805,180	823,240	1.67	*	818,840	818,840	0.00	173.6	818,840	818,840	0.00	163.7	818,840	818,840	0.00	**158.6**
10-5-4-T	1,117,723	1,147,530	2.31	*	1,144,160	1,144,160	0.00	141.4	1,144,160	1,144,160	0.00	**68.6**	1,144,160	1,144,160	0.00	72.6
15-10-4-T	1,575,640	1,605,460	1.34	*	1,597,100	1,597,100	0.00	**9.3**	1,597,100	1,597,100	0.00	11.5	1,597,100	1,597,100	0.00	13.7
20-10-3-T	1,551,597	-	4.78	*	1,629,000	1,629,500	**0.03**	*	1,629,000	1,629,500	**0.03**	*	1,629,000	1,629,500	**0.03**	*
20-12-3-T	1,541,949	1,628,900	4.02	*	1,606,500	1,606,500	0.00	46.3	1,606,500	1,606,500	0.00	45.4	1,606,500	1,606,500	0.00	**42.5**
Average			**1.113**				**0.0079**				**0.0060**				**0.0081**	
Optimal solutions			**15/34**				**30/34**				**30/34**				**30/34**	

Table 2.2: Computational results on instances from Venturini et al. (2017) with a total time limit of 3 hours and 18 minutes. The MIP formulation is compared to the variants of the presented branch-and-cut-and-price method. "*" means the time limit has been reached. The best running time is highlighted in bold for instances solved to optimality and the best optimality gap for the rest of instances.

Instance	MIP formulation				Branch & Price				Branch & Cut (root node) & Price				Branch & Cut & Price			
N-B-P-TW	LB	Z	Gap (%)	T (s)	LB	Z	Gap (%)	T (s)	LB	Z	Gap (%)	T (s)	LB	Z	Gap (%)	T (s)
4-3-3-L	296,600	296,600	0.00	**0.1**	296,600	296,600	0.00	0.5	296,600	296,600	0.00	0.2	296,600	296,600	0.00	0.2
5-3-3-L	394,300	394,300	0.00	**0.4**	394,300	394,300	0.00	3.2	394,300	394,300	0.00	7.2	394,300	394,300	0.00	7.2
6-3-3-L	421,679	421,720	0.01	2.3	421,720	421,720	0.00	0.8	421,720	421,720	0.00	**0.5**	421,720	421,720	0.00	**0.5**
6-3-4-L	647,423	647,480	0.01	89.2	647,480	647,480	0.00	**39.0**	647,480	647,480	0.00	111.7	647,480	647,480	0.00	103.7
10-4-4-L	1,020,581	1,055,800	3.22	*	1,054,500	1,054,500	0.00	**5563.8**	1,054,500	1,054,500	0.00	6068.2	1,054,300	1,054,500	**0.02**	*
10-4-3-L	694,451	699,000	0.52	*	698,100	698,100	0.00	193.6	698,100	698,100	0.00	**72.7**	698,100	698,100	0.00	115.6
4-4-4-L	405,120	405,120	0.00	**0.3**	405,120	405,120	0.00	0.6	405,120	405,120	0.00	0.6	405,120	405,120	0.00	0.6
5-4-4-L	500,600	500,600	0.00	**0.4**	500,600	500,600	0.00	0.9	500,600	500,600	0.00	0.8	500,600	500,600	0.00	0.8
6-4-4-L	599,980	599,980	0.00	**1.2**	599,980	599,980	0.00	7.6	599,980	599,980	0.00	5.1	599,980	599,980	0.00	5.3
12-5-3-L	813,713	834,740	2.01	*	830,440	830,440	0.00	136.0	830,440	830,440	0.00	**112.1**	830,440	830,440	0.00	120.4
10-6-3-L	680,600	680,600	0.00	219.4	680,600	680,600	0.00	9.7	680,600	680,600	0.00	**5.1**	680,600	680,600	0.00	5.8
11-6-3-L	746,220	746,220	0.00	8705.8	746,220	746,220	0.00	22.5	746,220	746,220	0.00	**12.7**	746,220	746,220	0.00	14.3
12-6-3-L	809,840	809,840	0.00	5032.0	809,840	809,840	0.00	112.6	809,840	809,840	0.00	79.0	809,840	809,840	0.00	**72.5**
10-5-4-L	1,013,114	1,029,300	1.48	*	1,028,320	1,028,320	0.00	589.0	1,028,320	1,028,320	0.00	**366.1**	1,028,320	1,028,320	0.00	1135.1
15-10-3-L	1,006,200	1,006,200	0.00	3259.5	1,006,200	1,006,200	0.00	46.4	1,006,200	1,006,200	0.00	27.8	1,006,200	1,006,200	0.00	**25.0**
15-12-3-L	1,002,240	1,002,800	0.06	*	1,002,800	1,002,800	0.00	**1.5**	1,002,800	1,002,800	0.00	**1.5**	1,002,800	1,002,800	0.00	**1.5**
15-10-4-L	1,459,600	1,459,600	0.00	1703.1	1,459,600	1,459,600	0.00	23.8	1,459,600	1,459,600	0.00	40.2	1,459,600	1,459,600	0.00	**13.5**
20-10-3-L	1,341,640	1,346,000	0.23	*	1,344,520	1,344,800	0.02	*	1,344,525	1,344,800	0.02	*	1,344,600	1,344,800	**0.01**	*
20-12-3-L	1,331,680	1,337,400	0.35	*	1,336,400	1,336,400	0.00	**2.1**	1,336,400	1,336,400	0.00	2.2	1,336,400	1,336,400	0.00	2.2
4-3-3-T	318,440	318,440	0.00	0.3	318,440	318,440	0.00	0.7	318,440	318,440	0.00	**0.2**	318,440	318,440	0.00	**0.2**
5-3-3-T	405,240	405,240	0.00	**0.6**	405,240	405,240	0.00	1.5	405,240	405,240	0.00	1.3	405,240	405,240	0.00	1.1
6-3-3-T	510,920	510,920	0.00	4.0	510,920	510,920	0.00	1.9	510,920	510,920	0.00	0.9	510,920	510,920	0.00	**0.8**
6-3-4-T	993,460	993,460	0.00	3.5	993,460	993,460	0.00	**1.2**	993,460	993,460	0.00	1.3	993,460	993,460	0.00	**1.2**
10-4-4-T	1,660,640	1,660,640	0.00	1660.0	1,660,640	1,660,640	0.00	101.3	1,660,640	1,660,640	0.00	63.9	1,660,640	1,660,640	0.00	**61.0**
10-4-3-T	1,022,200	1,022,200	0.00	562.5	1,022,200	1,022,200	0.00	12.3	1,022,200	1,022,200	0.00	**6.6**	1,022,200	1,022,200	0.00	7.9
4-4-4-T	442,600	442,600	0.00	0.9	442,600	442,600	0.00	1.1	442,600	442,600	0.00	0.7	442,600	442,600	0.00	**0.6**
5-4-4-T	576,010	576,010	0.00	**4.1**	576,010	576,010	0.00	10.3	576,010	576,010	0.00	6.0	576,010	576,010	0.00	6.6
6-4-4-T	653,560	653,560	0.00	11.5	653,560	653,560	0.00	23.5	653,560	653,560	0.00	**8.8**	653,560	653,560	0.00	10.4
12-5-3-T	817,533	830,440	1.55	*	830,440	830,440	0.00	128.0	830,440	830,440	0.00	**68.3**	830,440	830,440	0.00	96.1
12-6-3-T	810,476	821,540	1.02	*	818,840	818,840	0.00	173.6	818,840	818,840	0.00	163.7	818,840	818,840	0.00	**158.6**
10-5-4-T	1,144,160	1,144,160	0.00	3649.8	1,144,160	1,144,160	0.00	141.4	1,144,160	1,144,160	0.00	**68.6**	1,144,160	1,144,160	0.00	72.6
15-10-4-T	1,584,071	1,597,620	0.82	*	1,597,100	1,597,100	0.00	**9.3**	1,597,100	1,597,100	0.00	11.5	1,597,100	1,597,100	0.00	13.7
20-10-3-T	1,552,283	1,634,900	4.74	*	1,629,380	1,629,500	**0.01**	*	1,629,300	1,629,500	0.01	*	1,629,380	1,629,500	**0.01**	*
20-12-3-T	1,545,006	1,609,100	3.83	*	1,606,500	1,606,500	0.00	46.3	1,606,500	1,606,500	0.00	45.4	1,606,500	1,606,500	0.00	**42.5**
Average			**0.584**				**0.0008**				**0.0010**				**0.0012**	
Optimal solutions			**24/34**				**32/34**				**32/34**				**31/34**	

Table 2.3: Computational results on the set of harder instances with a total time limit of 5 minutes and 30 seconds. The MIP formulation is compared to the variants of the presented branch-and-cut-and-price method. "-" means that no integer solution has been found within the time limit. "*" means the time limit has been reached. The best running time is highlighted in bold for instances solved to optimality and the best optimality gap for the rest of instances.

Instance	**MIP formulation**				**Branch & Price**				**Branch & Cut (root node) & Price**				**Branch & Cut & Price**			
N-B-P-TW	**LB**	**Z**	**Gap (%)**	**T (s)**	**LB**	**Z**	**Gap (%)**	**T (s)**	**LB**	**Z**	**Gap (%)**	**T (s)**	**LB**	**Z**	**Gap (%)**	**T (s)**
25-12-3-L	1,668,080	-	0.56	*	1,677,200	1,677,400	**0.01**	*	1,677,200	1,677,400	**0.01**	*	1,677,200	1,677,400	**0.01**	*
25-12-3-T	1,923,560	-	6.03	*	2,046,930	2,047,100	0.00	*	2,046,933	2,047,200	**0.00**	*	2,046,933	2,047,200	**0.00**	*
12-5-4-L	1,201,546	1,253,160	3.50	*	1,239,277	1,245,360	0.47	*	1,240,826	1,247,760	0.35	*	1,240,862	1,247,060	**0.35**	*
12-5-4-T	1,324,428	-	5.69	*	1,398,848	1,410,270	0.39	*	1,399,631	1,408,070	**0.33**	*	1,398,580	1,408,220	0.41	*
30-12-3-L	1,997,400	-	0.95	*	2,016,178	2,016,600	**0.02**	*	2,016,178	2,016,600	**0.02**	*	2,016,178	2,016,600	**0.02**	*
30-12-3-T	2,304,432	-	7.55	*	2,491,406	2,496,900	**0.04**	*	2,491,406	2,495,300	**0.04**	*	2,491,406	2,495,600	**0.04**	*
20-12-4-L	1,934,640	-	0.47	*	1,943,486	1,943,800	0.02	*	1,943,500	1,943,800	0.02	*	1,943,800	1,943,800	0.00	**243.7**
20-12-4-T	3,050,995	-	2.00	*	3,113,170	3,113,170	0.00	42.5	3,113,170	3,113,170	0.00	15.8	3,113,170	3,113,170	0.00	**14.9**
15-8-4-L	1,471,903	1,508,500	1.68	*	1,495,225	1,497,100	0.12	*	1,496,131	1,497,300	**0.06**	*	1,496,118	1,497,100	0.06	*
15-8-4-T	1,599,496	-	3.41	*	1,654,848	1,656,260	0.07	*	1,655,376	1,656,260	0.04	*	1,655,416	1,656,260	**0.04**	*
25-12-4-L	2,419,800	-	0.85	*	2,439,377	2,440,700	0.05	*	2,439,531	2,440,600	**0.04**	*	2,439,380	2,441,500	0.05	*
25-12-4-T	3,560,100	-	3.97	*	3,706,995	3,707,390	0.01	*	3,707,182	3,707,390	0.01	*	3,707,390	3,707,390	**0.00**	*
30-15-4-L	2,905,000	-	0.47	*	2,918,400	2,918,800	0.01	*	2,918,420	2,918,800	0.01	*	2,918,436	2,918,800	**0.01**	*
30-15-4-T	3,109,816	-	5.14	*	3,274,118	3,278,880	0.13	*	3,274,390	3,283,550	0.12	*	3,274,441	3,280,400	**0.12**	*
40-15-3-L	2,658,400	-	1.15	*	2,689,060	2,689,200	**0.01**	*	2,689,060	2,689,200	**0.01**	*	2,689,060	2,689,200	**0.01**	*
40-15-3-T	3,067,200	-	7.89	*	3,329,567	3,330,900	**0.02**	*	3,329,567	3,330,900	**0.02**	*	3,329,567	3,331,300	**0.02**	*
Average			**3.206**				**0.0849**				**0.0665**				**0.0698**	
Optimal solutions			**0/16**				**1/16**				**1/16**				**3/16**	

Table 2.4: Computational results on the set of harder instances with a total time limit of 3 hours and 18 minutes. The MIP formulation is compared to the variants of the presented branch-and-cut-and-price method. "-" means that no integer solution has been found within the time limit. "*" means the time limit has been reached. The best running time is highlighted in bold for instances solved to optimality and the best optimality gap for the rest of instances.

Instance	**MIP formulation**				**Branch & Price**				**Branch & Cut (root node) & Price**				**Branch & Cut & Price**			
N-B-P-TW	**LB**	**Z**	**Gap (%)**	**T (s)**	**LB**	**Z**	**Gap (%)**	**T (s)**	**LB**	**Z**	**Gap (%)**	**T (s)**	**LB**	**Z**	**Gap (%)**	**T (s)**
25-12-3-L	1,668,080	1,679,400	0.56	*	1,677,200	1,677,400	**0.01**	*	1,677,200	1,677,400	**0.01**	*	1,677,200	1,677,400	**0.01**	*
25-12-3-T	1,932,150	-	5.61	*	2,046,930	2,047,100	0.00	*	2,046,933	2,047,100	**0.00**	*	2,046,933	2,047,000	**0.00**	*
12-5-4-L	1,205,096	1,253,160	3.22	*	1,244,427	1,245,660	0.06	*	1,245,110	1,245,160	**0.00**	*	1,244,080	1,245,160	0.09	*
12-5-4-T	1,348,176	1,407,980	4.00	*	1,404,280	1,404,280	0.00	**5126.1**	1,404,280	1,404,280	0.00	9530.6	1,403,243	1,405,760	0.07	*
30-12-3-L	1,998,483	-	0.90	*	2,016,178	2,016,600	**0.02**	*	2,016,178	2,016,600	**0.02**	*	2,016,178	2,016,600	**0.02**	*
30-12-3-T	2,311,705	-	7.25	*	2,491,406	2,492,500	**0.04**	*	2,491,406	2,492,500	**0.04**	*	2,491,406	2,492,500	**0.04**	*
20-12-4-L	1,934,640	-	0.47	*	1,943,800	1,943,800	0.00	2544.5	1,943,800	1,943,800	0.00	1202.7	1,943,800	1,943,800	0.00	**243.7**
20-12-4-T	3,055,040	-	1.87	*	3,113,170	3,113,170	0.00	42.5	3,113,170	3,113,170	0.00	15.8	3,113,170	3,113,170	0.00	**14.9**
15-8-4-L	1,475,646	1,497,000	1.43	*	1,497,000	1,497,000	0.00	2870.3	1,497,000	1,497,000	0.00	**968.3**	1,497,000	1,497,000	0.00	1831.1
15-8-4-T	1,608,977	1,671,220	2.84	*	1,656,040	1,656,040	0.00	**746.7.3**	1,656,040	1,656,040	0.00	1365.a	1,656,040	1,656,040	0.00	1534.0
25-12-4-L	2,420,034	-	0.84	*	2,439,594	2,440,500	0.04	*	2,439,796	2,440,500	0.03	*	2,439,881	2,440,500	**0.03**	*
25-12-4-T	3,572,971	-	3.63	*	3,707,390	3,707,390	0.00	604.3	3,707,390	3,707,390	0.00	501.1	3,707,390	3,707,390	0.00	**440.8**
30-15-4-L	2,905,000	-	0.47	*	2,918,400	2,918,800	0.01	*	2,918,425	2,918,800	0.01	*	2,918,441	2,918,600	**0.01**	*
30-15-4-T	3,123,445	-	4.72	*	3,274,905	3,278,280	0.10	*	3,275,095	3,278,280	0.10	*	3,275,112	3,278,940	**0.10**	*
40-15-3-L	2,658,440	-	1.14	*	2,689,060	2,689,200	**0.01**	*	2,689,060	2,689,200	**0.01**	*	2,689,060	2,689,200	**0.01**	*
40-15-3-T	3,071,346	-	7.77	*	3,329,567	3,330,300	**0.02**	*	3,329,567	3,330,200	**0.02**	*	3,329,567	3,330,100	**0.02**	*
Average			**2.919**				**0.0192**				**0.0148**				**0.0243**	
Optimal solutions			**0/16**				**6/16**				**6/16**				**5/16**	

Table 2.5: Performance of presented methods on a subset of five instances.

Instance	Branch & Price										
N-B-P-TW	Gap (%)	T (s)	T RMP (%)	T PP (%)	T Sep (%)	T Branch (%)	T GSPP (%)	Nodes	CG Its	Cols	Cuts
12-6-3-T	0.00	174	75.8	19.6	0.0	1.5	0.0	219	3,820	16,928	0
10-4-4-L	0.00	5,564	89.2	10.3	0.0	0.1	0.0	233	14,745	91,489	0
20-10-3-L	0.02	11,031	28.7	28.8	0.0	18.7	0.0	13,907	45,145	112,557	0
15-8-4-L	0.00	2,870	48.3	49.8	0.0	0.6	0.0	327	7,553	48,756	0
40-15-3-T	0.02	11,097	74.3	10.5	0.0	2.6	2.1	899	6,678	99,529	0
Instance	**Branch & Cut (root node) & Price**										
N-B-P-TW	Gap (%)	T (s)	T RMP (%)	T PP (%)	T Sep (%)	T Branch (%)	T GSPP (%)	Nodes	CG Its	Cols	Cuts
12-6-3-T	0.00	164	88.3	9.1	0.4	0.9	0.0	45	1,356	7,215	567
10-4-4-L	0.00	6,068	95.3	4.4	0.1	0.1	0.0	81	5,667	35,916	382
20-10-3-L	0.02	11,089	28.5	26.8	4.1	17.8	0.0	15,421	49,256	131,616	24
15-8-4-L	0.00	968	69.8	27.8	0.4	1.1	0.0	49	1,417	10,629	861
40-15-3-T	0.02	11,022	68.9	12.2	1.4	3.3	1.3	903	6,390	94,066	20
Instance	**Branch & Cut & Price**										
N-B-P-TW	Gap (%)	T (s)	T RMP (%)	T PP (%)	T Sep (%)	T Branch (%)	T GSPP (%)	Nodes	CG Its	Cols	Cuts
12-6-3-T	0.00	159	85.2	10.9	1.6	1.0	0.0	43	1,426	7,071	1,382
10-4-4-L	0.02	11,120	93.6	3.3	0.2	0.0	2.7	61	6,755	38,497	3,716
20-10-3-L	0.01	11,062	28.0	28.7	11.6	12.9	0.0	12,681	49,949	118,332	32,198
15-8-4-L	0.00	1,831	81.1	17.5	0.6	0.5	0.0	31	1,660	11,727	2,236
40-15-3-T	0.02	11,180	73.1	9.6	6.5	1.2	3.1	515	5,547	79,677	7,124

The results show a better performance of the solution methods based on the GSPP formulation. All variants of the *branch-and-cut-and-price* method are able to find optimal or near-optimal solutions in less than 6 minutes. Among the new proposed methods, the one where cutting is performed at the root node shows a better performance. All the solution method variants outperform CPLEX in all instances that require more than 4 seconds to solve and show similar running times for the faster ones. The difference in performance is more notable on the set of harder instances where CPLEX is not able to find a feasible integer solution in 11 out of the 16 instances within 3 hours and 18 minutes and the average optimality gap is above 2.9 %. Within 5 minutes and 30 seconds, the proposed new methods not only find feasible solutions to all instances but also achieve an optimality gap of 0.03 %. This gap is further reduced to less than 0.01 % with a time limit of 3 hours and 18 minutes. The good quality of the solutions in such a short computational time is attractive from an operational point of view where suboptimal solutions usually are not a problem and possible disruptions require rapid computations for new plans. Regarding the impact of solving the GSPP at the end, it is higher when the time limits are low. The GSPP is only solved if the method still has not proven optimality and, from those cases, it is found that the GSPP improves the upper bound in 84-100% of the cases with the 5 minutes and 30 seconds time limit and in 50-57% of the cases with the 3 hours and 18 minutes, depending on the method. In the vast majority of these cases, an integer solution is not found in the B&B tree.

Table 2.6: Optimality gap to optimal or best known solution at the root node.

Instance	MIP formulation		Without cuts		With sol-based cuts		With all cuts	
N-B-P-TW	**Gap (%)**	**T (s)**	**Gap (%)**	**T (s)**	**Gap (%)**	**T (s)**	**Gap (%)**	**T (s)**
12-6-3-T	2.03	0.3	0.36	2	0.22	5	0.22	87
10-4-4-L	4.69	0.1	0.22	15	0.15	102	0.15	1,137
20-10-3-L	0.36	0.2	0.03	2	0.03	2	0.03	143
15-8-4-L	2.11	0.2	0.25	19	0.12	43	0.12	499
40-15-3-T	7.89	1.2	0.02	280	0.02	285	0.02	685
Average 50 instances	**3.97**	**0.3**	**0.23**	**18**	**0.0865**	**25**	**0.0863**	**315**

Table 2.5 provides a more detailed comparison of the proposed method variants for 5 instances that aim to be representative of the entire set of 50 instances. The first column indicates the instance, the second and third column recap the optimality gap and computational time spent given the time-limit of 3 hours and 18 minutes. The fourth to eighth columns indicate the percentage amount of time spent by the algorithm in the RMP, pricing problems (PPs), cut separation process (Sep), branching procedure (Branch), and the final GSPP model respectively. The pricing problems are solved in parallel on the four cores used. It should also be noticed that the branching time not only includes the selection of the branching candidate but also, the child nodes creation, which in the case of our algorithm, requires intensive data structure manipulation. The number of nodes explored in the B&B tree is displayed in the ninth column. The last three columns indicate the number of column generation iterations, generated columns and added cuts respectively.

The RMP takes most of the time for most of the instances, and the cut separation has an insignificant impact except when it is applied in every B&B node. The time spent in branching procedures grows in accordance to both the size of the RMP and B&B tree. The short RMP solving times and large amount of columns generated for instance $20{-}10{-}3{-}L$ suggest that the RMP is easy to solve and the existence of many equivalent or similar solutions. This increases the impact of other internal operations in the algorithm. The number of B&B nodes explored grows inversely to the amount of nodes where cutting is allowed. It can be observed that the full *branch-and-cut-price* performs more column generation iterations than the one with only cutting in the root node but it also requires longer computational times. As it can be observed, the time percentages do not sum exactly to 100%. The remaining time accounts to diverse internal operations in the implementation which are not strictly linked to any of the main parts of the algorithm. This also suggests that there is room for improvement in the implementation of the algorithm.

The effectiveness of the aforementioned cut separation process is displayed in Table 2.6. The optimality gap of the LP solution at the root node is shown for the subset of five instances studied in detail together with the average across

Table 2.7: Solving time comparison between the network flow problem and the *branch-and-price* method.

Instance	**Graph size**		**Network flow problem**	**Branch & Price**
N-B-P-TW	**Nodes**	**Arcs**	**T(s)**	**T(s)**
4-3-3-L	1,621	44,989	5.4	0.5
5-3-3-L	2,711	1,669,872	383	3.2
6-3-3-L	2,711	2,353,886	517	0.8
6-3-4-L	5,414	8,659,488	2,504	39.0

all 50 instances. The second column denotes the LP solution at the root node for the MIP formulation. The third column refers to the presented method without adding any cuts whereas the fourth column considers the proposed cut separation procedure (Algorithm 2.2) based on solution values (i.e., *sol-based*). This procedure only checks a subset of the valid inequalities which we believe that contains most, if not all, of the violated ones. In any case, we can find all violated inequalities by simple enumeration. This case, where all violated valid inequalities (i.e., *all cuts*) are added, has also been tested and the results are shown in the last column. The improvement in the lower bound is significant for the proposed methods where the cut separation is able to further improve it achieving an average gap of 0.09 %. Adding all possible cuts only leads to an average improvement of 0.0002 % in the bound. However, the algorithm requires 12 times more computational time to solve the root node. It is therefore decided to discard this variant of the separation procedure given the slow performance and the insignificant gain.

As mentioned in section 2.4.1, when having a pure shortest path as a pricing problem, solving the LP relaxation of the network flow problem gives the same bound as column generation on the GSPP but the network flow problem is expected to require more time and memory resources on instances with dense graphs. In order to verify that, the network flow problem has been solved for the first four instances which are considered among the *easiest* ones from the entire set. The solving times of the network flow problem and the *branch-and-price* method are compared in Table 2.7. The complexity of the graph is shown by the high solving times for the network flow problem where the proposed model is between 10 and more than 500 times faster. The rest of instances have not been further analyzed as most of them were reaching the memory limit. The number of nodes and arcs for all the instances are documented in Appendix 2.A.3.

Apart from the presented methods, slight variations have been tested which helped to select the best algorithm procedure. For instance, we have tried to generate all columns *a priori* without success. The complexity of the problem

Table 2.8: Average performance of different branching strategies across all 50 instances using the *branch-and-cut-and-price* method with only cutting allowed in the root node.

Branching strategy	Best first on single node	Strong branching on berthing time	Best first on berthing position	Best first on berthing time
Average gap (%)	0.058	0.011	0.009	**0.005**
Optimal solutions	16/50	37/50	**39/50**	38/50

and the exponentially large numbers of columns make it intractable. Regarding branching procedures, an alternative method of exploring the B&B tree known as *strong branching* has been tested. This strategy requires to select a number of candidates (between 5 and 10 in our case) and compute, or at least estimate, the lower bounds at the child nodes. For each candidate, a weighted sum of the child bounds is computed and the candidate with the best weighted sum is selected. In this case, a weight of 0.75 is set for the child with the lowest bound and a weight of 0.25 for the other child. This method has proven to create better branches and, for example, is able to find an optimal solution to instance *20-10-3-T* in less than 20 minutes. Nevertheless, the overall worse optimality gap and additional time consumed exploring more nodes has lead us to discard it. A different branching strategy has been tested where the branching is done on berthing positions instead of on berthing times at a port. This strategy is able to solve all the instances from Venturini et al. (2017) to optimality and one more instance in overall than the presented method. However, the overall worse optimality gap indicates a poorer performance on the set of harder instances (see Appendix 2.A.3). In addition, a trivial branching on a single node has also been tested to compare the effectiveness of the proposed branching strategy. The solution values of the columns are added on the graph nodes of the respective paths, computing in this way the "usage" of each graph node. Then, the graph node whose value is closer to 0.5 (i.e., *most fractional*) is the one selected to branch on. A summary of the performance of these alternative branching strategies is displayed in Table 2.8 and the results for all instances can be found in Appendix 2.A.3.

2.6.2 Cooperative game theory results

The two methods for allocating the costs have been tested in the same set of instances. Three carriers A, B and C have been defined for all instances each of them with an assigned priority and a number of ships (see Table 2.9). This priority is often given in accordance to the handling volume (Imai et al., 2003). For instance, carrier A can be seen as a large carrier and often this translates in more power of decision and a higher priority at the port. The terminal operator at each visited port is also a player in the game. In this study, all ships visit

Table 2.9: Carrier ship share and priority for the instances.

Carrier	**A**	**B**	**C**
% of ships	50	25	25
Priority	1	2	3

Table 2.10: Terminal denomination.

Terminal	**D**	**E**	**F**	**G**
Visit position	1	2	3	4

the ports in the same order as shown in Figure 2.10, but the game can equally be applied to instances with different visit orders. Thus, depending on the instance's number of ports, the game is formed by either 6 or 7 players. The number of possible coalitions is given by $2^{|\mathcal{P}|}$ where $|\mathcal{P}|$ denotes the number of players, and in this case, corresponds to 64 or 128 coalitions respectively.

The overall cooperative game is based on what we denote as the *standalone solution*. This solution reflects the scenario where a single carrier negotiates with a single terminal at a time in order to decide the schedule for the carriers' vessels. We apply a greedy heuristic to compute this solution where we optimize and fix the schedule of a carrier's ships one port at a time. The sequence of carriers and ports used by the heuristic is given by the carrier's priority at port (see Table 2.9) and the position of the port visited (see Table 2.10). For example, assume ship 1 is carrier A's only ship and visits first port D and then port E where it has the highest priority. We then optimize the schedule of ship 1 for port D, fix the decisions and then optimize the schedule of ship 1 for port E. Once the port visits of a carrier's ships are scheduled, the ships of the next carrier with the highest priority are scheduled within the remaining available berthing positions and time windows.

This sequential planning approach resembles the actual procedure in some ports when the carriers book the port calls and they are assigned based on different priority schemes. Due to the heuristic nature of the process, some of the carriers may not find a feasible schedule. In order to avoid this and still ensure a fair comparison, the operational time windows of all berthing positions have been increased by 20% in the tests performed in this section.

The different coalitions $\mathcal{S} \subseteq \mathcal{P}$ can be classified into three groups, depending on the type of players forming it:

- Coalitions formed by carriers only. For the problem at hand this form of collaboration does not provide any planning advantage as the carriers

require collaboration with the terminal operators to improve their planning. Therefore, the solution of this type of coalition corresponds to the standalone solution.

- Coalitions formed by terminal operators only. For the problem at hand this form of collaboration does not provide any planning advantage as the terminal operators require collaboration with the carriers to improve their planning. Therefore, the solution of this type of coalition corresponds to the standalone solution.
- Coalitions formed by both carriers and terminal operators. This type of coalition is the basis for the MPBAP. In order to compute a solution to a given coalition, we assume that the planning of all players that are not part of the coalition are kept fixed as in the standalone solution. Then, the MPBAP is solved for the coalition given the available berthing space and time at the terminals. Note that when more players are part of the coalition, fewer port calls of the standalone solution need to be fixed. In addition, it can be noticed that optimal solutions to coalitions formed by a single carrier and a single operator are equivalent to the standalone solution. We believe this minimal collaboration resembles the real-life port call booking process for carriers.

The premise of fixing the port calls of non-collaborators, ensures that, in the worst case, the standalone solution is feasible for any coalition $\mathcal{S} \subseteq \mathcal{P}$. As indicated in Section 2.3.2, carriers and terminal operators have different operational costs. This is also reflected in the characteristic function, where the costs of each player are measured differently. On one side, the carrier's cost comprise the fuel consumption costs, waiting time costs and half of the delay costs. On the other side, the terminal operator's cost comprise the handling costs and the remaining half of the delay costs. The process of quantifying the delay costs in this type of problems is complex and it has been decided to equally split the delay costs between carriers and terminal operators.

As shown in Section 2.6.1, the grand coalition scenario for some instances are not solved to optimality, but the proposed methods are able to find solutions within a very small gap in less than 6 minutes (see Tables 2.1 and 2.3). It is assumed, that all the subcoalition scenarios are at most, as hard to solve as the grand coalition one and, therefore, a time limit of 5 minutes, and an additional 30 seconds to solve the GSPP, is set to solve each coalition scenario of the game.

All instances have a non-empty core, meaning that both efficient and stable solutions can be found in all scenarios. Tables 2.11 and 2.12 show the average cost allocations to each of the carriers and terminal operators across instances with 3 and 4 ports respectively (24 out of the 50 instances have 4 ports). For each allocation method we display three columns, (i) the first column indicates

Table 2.11: Comparison of the two cost allocation methods across instances with three ports.

	Player	Cost	Shapley value			Equal profit method (EPM)		
	$\mathcal{S}$	$\vartheta(\mathcal{S})$	f_i	Relative savings (%)	% of total costs	f_i	Relative savings (%)	% of total costs
Carrier	**A**	485,261	478,464	1.5	39.1	471,601	3.3	38.4
	B	247,053	242,800	2.2	20.0	240,685	3.1	19.8
	C	251,485	246,914	2.4	20.9	245,092	3.1	20.7
Terminal	**D**	88,635	79,042	10.7	7.0	86,058	3.4	7.5
	E	75,221	70,789	6.9	6.4	73,059	3.3	6.7
	F	75,296	71,546	5.8	6.6	73,060	3.2	6.8
	Grand coalition	1,189,555						

Table 2.12: Comparison of the two cost allocation methods across instances with four ports.

	Player	Cost	Shapley value			Equal profit method (EPM)		
	$\mathcal{S}$	$\vartheta(\mathcal{S})$	f_i	Relative savings (%)	% of total costs	f_i	Relative savings (%)	% of total costs
Carrier	**A**	352,695	347,232	1.7	37.7	339,450	4.0	36.8
	B	187,320	181,789	3.1	20.5	179,622	4.3	20.2
	C	192,634	187,148	3.1	20.9	184,727	4.4	20.6
Terminal	**D**	46,264	40,908	11.7	4.6	44,464	4.0	5.0
	E	44,079	38,207	14.1	4.1	42,215	4.4	4.6
	F	66,496	60,718	8.5	6.3	63,629	3.9	6.6
	G	60,346	55,765	7.6	5.9	57,659	3.9	6.1
	Grand coalition	911,766						

the cost allocation to the carrier when being part of the grand coalition, (ii) the second column computes the percentual savings compared with the player's standalone cost and (iii) the third column shows the percentage of the overall costs allocated to each player. Both allocation methods show that significant savings can be achieved by all of the players involved. In fact, player A, which in theory may be the least interested in engaging in such grand coalitions due to its high priority at all ports, achieves significant savings. The same applies to terminal D, which is the first one visited by the ships, and can benefit substantially by the overall better planning of the rest of terminals. The differences in the allocation strategy used by the two methods are noticeable. The EPM tends to equalize the relative savings of all players whereas the Shapley value is prone to balance the absolute savings.

As mentioned in Section 2.1, we conceive that the solution to the MPBAP and the cost allocation methods could be provided as a service by third party

software companies. To establish the side payments in practice, players would need to commit to the service for a pre-established period and agree with the potential savings estimated by the third party. This is required in order to define the number of participants in the collaboration. Moreover, to measure the savings of the MPBAP solution, we need an estimate of the standalone costs of each player in a non-collaborative scenario. The third party could estimate this cost for each player using the current planning software and we assume that the player would agree to that estimate. Once the agreement is in place, the third party could be used as a proxy for the side payments, which could be performed on a regular basis. Based on the actual costs, each player would need to make or receive a payment to align with the projected savings that the player has agreed to.

Similar collaboration is already taking place in real-life in tramp shipping where it is common that a number of ship owners place their ships in a shipping pool under the control of a pool administrator (the analog to the third party coordinator) that takes over most of the business decisions regarding the ships and is responsible for distributing earnings to ship owners (Packard, 1995; Wen et al., 2019).

2.7 Conclusions and future work

A novel solution method based on a GSPP formulation has been presented for the MPBAP. The method exploits a graph formulation for defining the berthing plan of a ship along its route. This, combined together with delayed column generation, additional valid inequalities and symmetry breaking constraints results in an efficient algorithm able to find optimal or near-optimal solutions to wide range of instances outperforming the capacity of commercial solvers.

In addition, the graph formulation adds flexibility as many additional constraints can be easily integrated with simple alterations in the graph. For instance, a finer discretization of the berthing positions would allow to approximate the continuous version of the MPBAP better. Considering a continuous berth is a more realistic approach and allows to increase the usage of the quay. Transhipments are also an important aspect of the operations at port and the fulfillment of them are crucial in some cases (e.g., when transporting perishable food). Our model could eventually account for that by limiting the time window of the ships involved in the transhipment and penalizing late arrivals of the incoming ship or too early departures of the outgoing ship. Nevertheless, this could be better modelled if the relative arrival and departure times are considered. That case is harder to incorporate in the presented model and it would require additional constraints for each transhipment in the RMP. The transit times between ports

could be further improved by considering the time needed to enter and leave the port which is usually performed at a slower speed (Reinhardt et al., 2016).

The instances solved reflect the size of real-life scenarios to a large extent. However, some of the instance parameters could be further improved. This comprises improving the size of the vessel time windows, having different routes for the ships, different amount of berthing positions per port and different ship types.

Alternative branching methods have also shown great potential, especially branching on berthing positions as opposed to branching on berthing times. A natural next step would be to explore a branching method that combines both of them. For instance, one could test both methods simultaneously when branching and select the one with better bounds at the children nodes.

A natural extension of the problem could be to integrate the berth allocation with the quay crane assignment problem (QAP). Studies such as Iris et al. (2015) and Iris et al. (2017) have already shown the effectiveness of heuristic and exact methods based on a GSPP formulation for the integrated problem in one terminal.

Last but not least, the benefits for both ship carriers and terminal operators are verified defining a cooperative game and using cost allocation methods to distribute the costs of such collaboration fairly. The results of the cooperative game strengthen the viability of such a decision tool and can encourage carriers and port operators to participate in collaborative schemes.

Acknowledgement: This work is partially supported by The Danish Maritime Fund. The authors are grateful to Dr. Cagatay Iris for the useful discussion about the model and instances and to Prof. Harilaos Psaraftis and Dr. Thalis Zis for their valuable insights. Also, the authors are thankful to the associate editor and to two anonymous reviewers for their constructive remarks that have helped improve the manuscript.

References

APM Terminals (2021). APM Terminals. https://www.apmterminals.com/. Accessed: 2021-09-15.

Bektaş, T., Ehmke, J. F., Psaraftis, H. N., and Puchinger, J. (2019). The role of operational research in green freight transportation. *European Journal of Operational Research*, 274(3):807–823.

Bezanson, J., Edelman, A., Karpinski, S., and Shah, V. B. (2017). Julia: A fresh approach to numerical computing. *Siam Review*, 59(1):65–98.

Bierwirth, C. and Meisel, F. (2015). A follow-up survey of berth allocation and quay crane scheduling problems in container terminals. *European Journal of Operational Research*, 244(3):12689, 675–689.

Brouer, B. D., Pisinger, D., and Spoorendonk, S. (2011). Liner shipping cargo allocation with repositioning of empty containers. *INFOR Journal*, 49(2):109–124.

Buhrkal, K. F., Zuglian, S., Røpke, S., Larsen, J., and Lusby, R. M. (2011). Models for the discrete berth allocation problem: A computational comparison. *Transportation Research. Part E: Logistics and Transportation Review*, 47(4):461–473.

Carlo, H. J., Vis, I. F., and Roodbergen, K. J. (2014). Transport operations in container terminals: Literature overview, trends, research directions and classification scheme. *European Journal of Operational Research*, 236(1):1–13.

Chang, Y. T., Tongzon, J., Luo, M., and Lee, P. T. W. (2012). Estimation of optimal handling capacity of a container port: An economic approach. *Transport Reviews*, 32(2):241–258.

Cheong, C. Y., Tan, K. C., Liu, D. K., and Lin, C. J. (2010). Multi-objective and prioritized berth allocation in container ports. *Annals of Operations Research*, 180(1):63–103.

Cordeau, J. F., Laporte, G., Legato, P., and Moccia, L. (2005). Models and tabu search heuristics for the berth-allocation problem. *Transportation Science*, 39(4):526–538.

Cormen, T., Leiserson, C., and Rivest, R. (1996). *Introduction to algorithms*. MIT Press,.

Corry, P. and Bierwirth, C. (2019). The berth allocation problem with channel restrictions. *Transportation Science*, 53(3):708–727.

Dantzig, G. and Wolfe, P. (1960). Decomposition principle for linear-programs. *Operations Research*, 8(1):101–111.

Du, Y., Chen, Q., Lam, J. S. L., Xu, Y., and Cao, J. X. (2015). Modeling the impacts of tides and the virtual arrival policy in berth allocation. *Transportation Science*, 49(4):939–956.

Du, Y., Chen, Q., Quan, X., Long, L., and Fung, R. Y. (2011). Berth allocation considering fuel consumption and vessel emissions. *Transportation Research Part E: Logistics and Transportation Review*, 47(6):1021–1037.

Dulebenets, M. A. (2018). A comprehensive multi-objective optimization model for the vessel scheduling problem in liner shipping. *International Journal of Production Economics*, 196:293–318.

Dulebenets, M. A. (2019). Minimizing the total liner shipping route service costs via application of an efficient collaborative agreement. *Ieee Transactions on Intelligent Transportation Systems*, 20(1):8315131.

Dulebenets, M. A., Golias, M. M., and Mishra, S. (2018). A collaborative agreement for berth allocation under excessive demand. *Engineering Applications of Artificial Intelligence*, 69:76–92.

Dulebenets, M. A., Pasha, J., Abioye, O. F., and Kavoosi, M. (2019). Vessel scheduling in liner shipping: a critical literature review and future research needs. *Flexible Services and Manufacturing Journal*, 33(1):43–106.

Dunning, I., Huchette, J., and Lubin, M. (2017). Jump: A modeling language for mathematical optimization. *SIAM Review*, 59(2):295–320.

Ergun, O., Kuyzu, G., and Savelsbergh, M. (2007). Reducing truckload transportation costs through collaboration. *Transportation Science*, 41(2):206–221.

Fagerholt, K. (2001). Ship scheduling with soft time windows: An optimisation based approach. *European Journal of Operational Research*, 131(3):559–571.

Fagerholt, K., Gausel, N. T., Rakke, J. G., and Psaraftis, H. N. (2015). Maritime routing and speed optimization with emission control areas. *Transportation Research, Part C: Emerging Technologies*, 52:57–73.

Fagerholt, K., Laporte, G., and Norstad, I. (2010). Reducing fuel emissions by optimizing speed on shipping routes. *Journal of the Operational Research Society*, 61(3):523–529.

Fagerholt, K. and Psaraftis, H. N. (2015). On two speed optimization problems for ships that sail in and out of emission control areas. *Transportation Research. Part D: Transport and Environment*, 39:56–64.

Frisk, M., Göthe-Lundgren, M., Jörnsten, K., and Rönnqvist, M. (2010). Cost allocation in collaborative forest transportation. *European Journal of Operational Research*, 205(2):448–458.

Guan, Y. and Cheung, R. K. (2005). The berth allocation problem: Models and solution methods. *Container Terminals and Automated Transport Systems: Logistics Control Issues and Quantitative Decision Support*, pages 141–158.

Hansen, P. and Oguz, C. (2003). A note on formulation of the static and dynamic berth allocation problems. *Les Cahiers Du Gerad*, 30:1–17.

Imai, A., Nishimura, E., and Papadimitriou, S. (2003). Berth allocation with service priority. *Transportation Research Part B: Methodological*, 37(5):437–457.

Imai, A., Sun, X., Nishimura, E., and Papadimitriou, S. (2005). Berth allocation in a container port: using a continuous location space approach. *Transportation Research Part B-methodological*, 39(3):199–221.

IMO (2018). Initial IMO strategy on reduction of GHG emissions from ships. Technical Report MEPC.304(72), International Maritime Organization. (Accessed on 04.05.2020).

IMO (2020). Reduction of GHG emissions from ships. Fourth IMO GHG study 2020. Technical Report MEPC.304(72), International Maritime Organization. (Accessed on 22.08.2020).

Iris, C., Pacino, D., and Røpke, S. (2017). Improved formulations and an adaptive large neighborhood search heuristic for the integrated berth allocation and quay crane assignment problem. *Transportation Research. Part E: Logistics and Transportation Review*, 105:123–147.

Iris, C., Pacino, D., Røpke, S., and Larsen, A. (2015). Integrated berth allocation and quay crane assignment problem: Set partitioning models and computational results. *Transportation Research. Part E: Logistics and Transportation Review*, 81:75–97.

Jans, R. (2010). Classification of dantzig-wolfe reformulations for binary mixed integer programming problems. *European Journal of Operational Research*, 204(2):251–254.

Kontovas, C. and Psaraftis (2011). Reduction of emissions along the maritime inter modal container chain: Operational models and policies. *Maritime Policy and Management*, 38(4):451–469.

Kordić, S., Davidović, T., Kovač, N., and Dragović, B. (2016). Combinatorial approach to exactly solving discrete and hybrid berth allocation problem. *Applied Mathematical Modelling*, 40(21-22):8952–8973.

Krajewska, M. A., Kopfer, H., Laporte, G., Ropke, S., and Zaccour, G. (2008). Horizontal cooperation among freight carriers: request allocation and profit sharing. *Journal of the Operational Research Society*, 59(11):1483–1491.

Kramer, A., Lalla-Ruiz, E., Iori, M., and Voß, S. (2019). Novel formulations and modeling enhancements for the dynamic berth allocation problem. *European Journal of Operational Research*, 278(1):170–185.

Lalla-Ruiz, E., Expósito-Izquierdo, C., Melián-Batista, B., and Moreno-Vega, J. M. (2016a). A set-partitioning-based model for the berth allocation problem under time-dependent limitations. *European Journal of Operational Research*, 250(3):1001–1012.

Lalla-Ruiz, E., Melián-Batista, B., and Moreno-Vega, J. M. (2016b). A cooperative search for berth scheduling. *Knowledge Engineering Review*, 31(05):498–507.

Lim, A. (1998). The berth planning problem. *Operations Research Letters*, 22(2-3):105–110.

Maersk (2021). Maersk. https://www.maersk.com/. Accessed: 2021-09-15.

Magnanti, T. L., Orlin, J. B., and Ahuja, R. K. (1993). *Network flows : theory, algorithms, and applications*. Prentice-Hall.

Meisel, F. and Bierwirth, C. (2009). Heuristics for the integration of crane productivity in the berth allocation problem. *Transportation Research Part E: Logistics and Transportation Review*, 45(1):196–209.

Meng, Q. and Wang, S. (2011). Optimal operating strategy for a long-haul liner service route. *European Journal of Operational Research*, 215(1):105–114.

Navis (2021). Navis. https://www.navis.com/. Accessed: 2021-09-07.

Notteboom, T. E., Parola, F., Satta, G., and Pallis, A. A. (2017). The relationship between port choice and terminal involvement of alliance members in container shipping. *Journal of Transport Geography*, 64:158–173.

Notteboom, T. E. and Vernimmen, B. (2009). The effect of high fuel costs on liner service configuration in container shipping. *Journal of Transport Geography*, 17(5):325–337.

Özener, O. O., Ergun, O., and Savelsbergh, M. (2011). Lane-exchange mechanisms for truckload carrier collaboration. *Transportation Science*, 45(1):1–17.

Packard, W. (1995). *Shipping Pools*. Business of Shipping S. Lloyd's of London Press.

Pang, K. W. and Liu, J. (2014). An integrated model for ship routing with transshipment and berth allocation. *Iie Transactions (institute of Industrial Engineers)*, 46(12):1357–1370.

Portchain (2021). https://www.portchain.com. Accessed: 2021-09-06.

Psaraftis, H. N. and Kontovas, C. A. (2013). Speed models for energy-efficient maritime transportation: A taxonomy and survey. *Transportation Research Part C-emerging Technologies*, 26:331–351.

Psaraftis, H. N. and Kontovas, C. A. (2015a). Green maritime transportation: Speed and route optimization. *International Series in Operations Research and Management Science*, 226:299–349.

Psaraftis, H. N. and Kontovas, C. A. (2015b). Slow steaming in maritime transportation: Fundamentals, trade-offs, and decision models. *Handbook of Ocean Container Transport Logistics*, pages 315–358.

Pujats, K., Golias, M., and Konur, D. (2020). A review of game theory applications for seaport cooperation and competition. *Journal of Marine Science and Engineering*, 8(2):100.

RBS (2021). Realtime Business Solutions. https://www.rbs-tops.com/. Accessed: 2021-09-06.

Reinhardt, L. B., Plum, C. E., Pisinger, D., Sigurd, M. M., and Vial, G. T. (2016). The liner shipping berth scheduling problem with transit times. *Transportation Research. Part E: Logistics and Transportation Review*, 86:116–128.

Saadaoui, Y., Umang, N., and Frejinger, E. (2015). *A column generation framework for berth scheduling at port terminals*. CIRRELT, Centre interuniversitaire de recherche sur les réseaux d'entreprise ....

Saeed, N. and Larsen, O. I. (2010). An application of cooperative game among container terminals of one port. *European Journal of Operational Research*, 203(2):393–403.

Sealytix (2021). Sealytix. https://www.sealytix.com/. Accessed: 2021-09-20.

Shapley, L. S. (1953). A value for n-person games. *Contributions to the Theory of Games*, 2(28):307–317.

Ship & Bunker (2021). Global Average Bunker Price - VLSFO. https://shipandbunker.com/prices/av/global/av-glb-global-average-bunker-price#VLSFO. Accessed: 2021-03-21.

Song, D. W. (2003). Port co-opetition in concept and practice. *Maritime Policy and Management*, 30(1):29–44.

Song, D. W. and Panayides, P. M. (2002). A conceptual application of cooperative game theory to liner shipping strategic alliances. *Maritime Policy and Management*, 29(3):285–301.

Steenken, D., Voß, S., and Stahlbock, R. (2004). Container terminal operation and operations research - a classification and literature review. *Or Spectrum*, 26(1):3–49.

Sun, B., Niu, B., Xu, H., and Ying, W. (2018). Cooperative optimization for port and shipping line with unpredictable disturbance consideration. *Icnc-fskd 2018 - 14th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery*, pages 8686901, 113–118.

TGI (2021). TGI Maritime Software. https://www.tgims.com/. Accessed: 2021-09-06.

UNCTAD (2019). *Review of Maritime Transport 2019*. (Accessed on 27.06.2020).

Venturini, G., Iris, C., Kontovas, C. A., and Larsen, A. (2017). The multi-port berth allocation problem with speed optimization and emission considerations. *Transportation Research. Part D: Transport and Environment*, 54:142–159.

Wang, H., Wang, S., and Meng, Q. (2014). Simultaneous optimization of schedule coordination and cargo allocation for liner container shipping networks. *Transportation Research Part E: Logistics and Transportation Review*, 70(1):261–273.

Wang, S., Liu, Z., and Qu, X. (2015). Collaborative mechanisms for berth allocation. *Advanced Engineering Informatics*, 29(3):572, 332–338.

Wang, S. and Meng, Q. (2012). Liner ship route schedule design with sea contingency time and port time uncertainty. *Transportation Research Part B*, 46(5):615–633.

Wang, S., Meng, Q., and Liu, Z. (2013a). Bunker consumption optimization methods in shipping: A critical review and extensions. *Transportation Research Part E: Logistics and Transportation Review*, 53(1):49–62.

Wang, S., Meng, Q., and Liu, Z. (2013b). A note on "berth allocation considering fuel consumption and vessel emissions". *Transportation Research Part E: Logistics and Transportation Review*, 49(1):48–54.

Wen, M., Larsen, R., Røpke, S., Petersen, H. L., and Madsen, O. B. (2019). Centralised horizontal cooperation and profit sharing in a shipping pool. *Journal of the Operational Research Society*, 70(5):737–750.

Wen, M., Pacino, D., and Kontovas, C. (2017). A multiple ship routing and speed optimization problem under time, cost and environmental objectives. *Transportation Research. Part D: Transport and Environment*, 52(A):303–321.

World Shipping Council (2015). Some observations on port congestion, vessel size and vessel sharing agreements. Technical report, World Shipping Council. (Accessed on 07.03.2021).

Zhen, L., Hu, Z., Yan, R., Zhuge, D., and Wang, S. (2020). Route and speed optimization for liner ships under emission control policies. *Transportation Research Part C: Emerging Technologies*, 110:330–345.

2.A Appendix

2.A.1 Reduced cost computation including valid inequalities (2.33)

We denote $\beta^{n,p,b}_{t_1,t_2}$ to the dual variable of constraint (2.33) for ship $n \in N$, port $p \in P$, berth $b \in B_p$ and times $t_1, t_2 \in [s^{p,b}; e^{p,b}], t_1 < t_2$ and let $\bar{\beta}^{n,p,b}_{t_1,t_2}$ be its value for the RMP solution. Let $w(p,b,t) \in O$ be the graph node related to berthing at port $p \in P$ at position $b \in B_p$ at time $t \in [s^{p,b}; e^{p,b}]$ and let $\Phi = \{(n,p,b,t_1,t_2)\}$ be the set of constraints (33) added to the RMP denoted by (n,p,b,t_1,t_2) elements where $n \in N$, $p \in P$, $b \in B_p$ and $t_1, t_2 \in [s^{p,b}; e^{p,b}], t_1 < t_2$. Additionally, let $\Phi(k,p,b,t) \subseteq \Phi$ be the set of valid inequalities that include arcs from the graph node $w(p,b,t)$ for a given ship k. By definition the range of nodes for each ship within a valid inequality differ if $k = n$ or $k \neq n$. Therefore we denote $\Phi_{k=n}(k,p,b,t), \Phi_{k\neq n}(k,p,b,t) \subseteq \Phi(k,p,b,t)$ to the subset of cuts (n,p,b,t_1,t_2) where $k = n$ and $k \neq n$ respectively, that together form the entire set $\Phi(k,p,b,t) = \Phi_{k=n}(k,p,b,t) \cup \Phi_{k\neq n}(k,p,b,t)$ and are defined mathematically as follows:

$$\Phi_{k=n}(k,p,b,t) = \Big\{(n,p,b,t_1,t_2) | k = n, w(p,b,t) \in \bigcup_{t\in[t_1;t_2]} C(k,p,b,t)\Big\}$$

$$\Phi_{k\neq n}(k,p,b,t) = \Big\{(n,p,b,t_1,t_2) | k \neq n, w(p,b,t) \in C(k,p,b,t_1) \cap C(k,p,b,t_2)\Big\}$$

The computation of the reduced cost $\hat{c}_j$ for column j of ship $k \in N$ is updated as follows:

$$\hat{c}_j = c_j - \Big(\sum_{(p,b,t)\in\Lambda_j} \big(\sum_{t'\in[t;t+h^{p,b}_k)} \bar{\mu}_{p,b,t'} \big) - \big(\sum_{(n,p,b,t_1,t_2)\in\Phi(k,p,b,t)} \bar{\beta}^{n,p,b}_{t_1,t_2} \big) \Big) - \bar{\alpha}_k$$

2.A.2 Adaption of the proposed valid inequality considering berth types

Proposition 2.2 *Given two time instants $t_1, t_2 \in [s^{p,k}, e^{p,k})$ where $t_1 < t_2$ and a port $p \in P$, berth type $k \in K_p$ and ship $n \in N$, the following is a valid inequality:*

$$\sum_{u\in\bigcup_{t\in[t_1;t_2]} C(n,p,k,t)} \sum_{w\in\delta^+_n(u)} x^n_{u,w} + \sum_{m\in N\setminus\{n\}} \sum_{u\in C(m,p,k,t_1)\cap C(m,p,k,t_2)} \sum_{w\in\delta^+_m(u)} x^m_{u,w} \leqslant \beta^k$$

Proof.Constraint (34) has been adapted from constraint (30) which is a direct translation from constraint (26) from the network-flow formulation. Therefore,

constraint (34) can be formulated as a network-flow problem constraint as follows:

$$\sum_{m\in N}\sum_{i\in C(m,p,k,t)}\sum_{j\in\delta_m^+(i)} x_{i,j}^m \leq \beta^k \quad \forall p\in P, k\in K_p, t\in[s^{p,k};e^{p,k})$$

where the $C(m,p,k,t)$ defines the set of nodes for ship $m \in N$ that are in conflict with time $t \in [s^{p,k}; e^{p,k})$ in berth type $k \in K_p$ of port $p \in P$. Based on this definition, the intersection set $C(m,p,k,t_1) \cap C(m,p,k,t_2)$ directly defines the set of nodes for ship m that are in conflict with both time instants t_1 and t_2. Constraint (34) indicates that at most β^k (i.e., number of berths of type $k \in B_k$) arcs can be chosen out of the nodes from the sets $C(m,p,k,t)$ of all ships $m \in N$ and, therefore, the same applies to the intersection sets $C(m,p,k,t_1) \cap C(m,p,k,t_2)$. Based on the premise that each ship can only berth in one position, we can relax the requirement of being in conflict with both t_1 and t_2 for a single ship n and only require it to be in conflict with t_1 or t_2. In practice, this means, on one hand, that if ship n berths at a period covering t_1 or t_2, then, at most $\beta^k - 1$ ships $m \in N\backslash\{n\}$ can berth in a period covering both t_1 and t_2. On the other hand, if β^k ships $m \in N\backslash\{n\}$ are berthing at times whose periods cover t_1 and t_2, then ship n is not able to berth at a period covering t_1 or t_2. The relaxed node interval for ship n can be defined as the union of $C(n,p,k,t_1) \cup C(n,p,k,t_2)$. Considering these node sets, we can define the following valid inequality:

$$\sum_{u\in C(n,p,k,t_1)\cup C(n,p,k,t_2)}\sum_{w\in\delta_n^+(u)} x_{u,w}^n + \sum_{m\in N\backslash\{n\}}\sum_{u\in C(m,p,k,t_1)\cap C(m,p,k,t_2)}\sum_{w\in\delta_m^+(u)} x_{u,w}^m \leqslant \beta^k$$
$$\forall p\in P, k\in K_p, n\in N, t_1,t_2\in[s^{p,k},e^{p,k}), t_1<t_2$$

Based on the assumption that a berthing period cannot be discontinued, the intersection set $C(m,p,k,t_1) \cap C(m,p,k,t_2)$ for any ship is not only in conflict with times t_1 and t_2 but with all the time instants in the period $[t_1;t_2]$. Therefore the interval for ship n can be expanded to the union of $C(n,p,k,t)$ sets for all time instants $t \in [t_1;t_2]$ and the resulting valid inequality can be formulated as follows:

$$\sum_{u\in\bigcup_{t\in[t_1;t_2]} C(n,p,k,t)}\sum_{w\in\delta_n^+(u)} x_{u,w}^n + \sum_{m\in N\backslash\{n\}}\sum_{u\in C(m,p,k,t_1)\cap C(m,p,k,t_2)}\sum_{w\in\delta_m^+(u)} x_{u,w}^m \leqslant \beta^k$$
$$\forall p\in P, k\in K_p, n\in N, t_1,t_2\in[s^{p,k},e^{p,k}), t_1<t_2$$

□

2.A.3 Additional computational results

Table 2.13: Number of nodes and arcs in graph G for each of the instances. An horizontal line is used to indicate the separation between the set of benchmark instances by Venturini et al. (2017) and the newly generated set of harder instances.

Instance	Graph size		Instance	Graph size	
N-B-P-TW	**Nodes**	**Arcs**	**N-B-P-TW**	**Nodes**	**Arcs**
4-3-3-L	1,621	44,989	4-4-4-T	3,948	1,239,853
5-3-3-L	2,711	1,669,872	5-4-4-T	3,948	1,498,010
6-3-3-L	2,711	2,353,886	6-4-4-T	3,948	1,318,353
6-3-4-L	5,414	8,659,488	12-5-3-T	3,217	2,572,191
10-4-4-L	7,218	25,069,050	12-6-3-T	3,860	3,586,436
10-4-3-L	3,614	6,583,065	10-5-4-T	5,722	8,069,850
4-4-4-L	6,418	5,631,290	15-10-4-T	8,870	25,003,591
5-4-4-L	6,418	8,436,182	20-10-3-T	5,120	11,456,682
6-4-4-L	6,418	10,062,500	20-12-3-T	6,826	21,299,213
12-5-3-L	4,517	12,072,289	25-12-3-L	7,226	65,709,632
10-6-3-L	5,420	15,698,842	25-12-3-T	6,826	26,586,128
11-6-3-L	5,420	15,640,325	12-5-4-L	9,022	45,812,954
12-6-3-L	5,420	17,117,528	12-5-4-T	5,722	9,550,769
10-5-4-L	9,022	38,507,345	30-12-3-L	7,226	78,917,800
15-10-3-L	5,420	21,596,125	30-12-3-T	6,826	31,947,951
15-12-3-L	7,226	37,072,022	20-12-4-L	15,638	171,445,244
15-10-4-L	12,430	76,196,427	20-12-4-T	11,158	56,721,679
20-10-3-L	5,420	28,790,640	15-8-4-L	14,434	144,240,102
20-12-3-L	7,226	52,608,061	15-8-4-T	9,154	29,624,264
4-3-3-T	1,621	229,372	25-12-4-L	15,638	214,550,093
5-3-3-T	2,711	2,696,410	25-12-4-T	11,158	70,841,345
6-3-3-T	2,711	2,538,662	30-15-4-L	16,541	300,541,288
6-3-4-T	3,464	2,307,063	30-15-4-T	11,801	98,992,726
10-4-4-T	4,618	6,633,974	40-15-3-L	8,129	133,728,684
10-4-3-T	2,614	1,825,953	40-15-3-T	7,679	54,138,993

Table 2.14: Results of solution methods with alternative branching strategies. The underlying algorithm is a *branch-and-cut-and-price* where cutting is only allowed at the root node. "*" means the time limit of 3 hours and 18 minutes has been reached.

Instance	Best first on single graph node				Strong branching on berthing time				Best first on berthing position			
N-B-P-TW	**LB**	**Z**	**Gap (%)**	**T (s)**	**LB**	**Z**	**Gap (%)**	**T (s)**	**LB**	**Z**	**Gap (%)**	**T (s)**
4-3-3-L	296,600	296,600	0.00	0.2	296,600	296,600	0.00	0.2	296,600	296,600	0.00	0.2
5-3-3-L	394,300	394,300	0.00	8.9	394,300	394,300	0.00	9.2	394,300	394,300	0.00	8.1
6-3-3-L	421,720	421,720	0.00	0.5	421,720	421,720	0.00	0.6	421,720	421,720	0.00	0.5
6-3-4-L	647,149	647,480	0.05	*	647,480	647,480	0.00	1,024.4	647,480	647,480	0.00	93.7
10-4-4-L	1,053,050	1,054,500	0.14	*	1,054,250	1,054,500	0.02	*	1,054,500	1,054,500	0.00	9,628.2
10-4-3-L	697,057	698,100	0.15	*	698,100	698,100	0.00	443.5	698,100	698,100	0.00	173.0
4-4-4-L	405,120	405,120	0.00	0.9	405,120	405,120	0.00	0.7	405,120	405,120	0.00	0.6
5-4-4-L	500,600	500,600	0.00	1.2	500,600	500,600	0.00	0.9	500,600	500,600	0.00	0.9
6-4-4-L	599,780	599,980	0.03	*	599,980	599,980	0.00	26.9	599,980	599,980	0.00	7.4
12-5-3-L	829,897	830,440	0.07	*	830,440	830,440	0.00	646.1	830,440	830,440	0.00	143.3
10-6-3-L	680,550	680,600	0.01	*	680,600	680,600	0.00	34.4	680,600	680,600	0.00	15.1
11-6-3-L	745,960	746,220	0.03	*	746,220	746,220	0.00	52.4	746,220	746,220	0.00	16.9
12-6-3-L	809,093	810,040	0.09	*	809,840	809,840	0.00	301.7	809,840	809,840	0.00	54.3
10-5-4-L	1,026,521	1,028,320	0.17	*	1,028,320	1,028,320	0.00	2,472.5	1,028,320	1,028,320	0.00	5,991.3
15-10-3-L	1,006,000	1,006,200	0.02	*	1,006,200	1,006,200	0.00	59.8	1,006,200	1,006,200	0.00	4.8
15-12-3-L	1,002,800	1,002,800	0.00	2.0	1,002,800	1,002,800	0.00	1.8	1,002,800	1,002,800	0.00	2.3
15-10-4-L	1,459,600	1,459,600	0.00	339.9	1,459,600	1,459,600	0.00	62.6	1,459,600	1,459,600	0.00	14.3
20-10-3-L	1,344,437	1,344,800	0.03	*	1,344,583	1,344,800	0.02	*	1,344,800	1,344,800	0.00	163.3
20-12-3-L	1,336,400	1,336,400	0.00	3.4	1,336,400	1,336,400	0.00	2.9	1,336,400	1,336,400	0.00	2.2
4-3-3-T	318,440	318,440	0.00	0.5	318,440	318,440	0.00	0.3	318,440	318,440	0.00	0.3
5-3-3-T	405,240	405,240	0.00	2.5	405,240	405,240	0.00	3.6	405,240	405,240	0.00	7.4
6-3-3-T	510,920	510,920	0.00	1.5	510,920	510,920	0.00	1.1	510,920	510,920	0.00	0.8
6-3-4-T	993,460	993,460	0.00	2.5	993,460	993,460	0.00	1.8	993,460	993,460	0.00	1.2
10-4-4-T	1,660,640	1,660,640	0.00	1,193.6	1,660,640	1,660,640	0.00	329.9	1,660,640	1,660,640	0.00	132.9
10-4-3-T	1,022,200	1,022,200	0.00	49.1	1,022,200	1,022,200	0.00	56.8	1,022,200	1,022,200	0.00	9.1
4-4-4-T	442,600	442,600	0.00	1.3	442,600	442,600	0.00	1.0	442,600	442,600	0.00	0.8
5-4-4-T	575,350	576,010	0.11	*	576,010	576,010	0.00	26.7	576,010	576,010	0.00	9.6
6-4-4-T	651,480	654,040	0.32	*	653,560	653,560	0.00	39.0	653,560	653,560	0.00	10.6
12-5-3-T	830,067	830,440	0.04	*	830,440	830,440	0.00	355.7	830,440	830,440	0.00	100.0
12-6-3-T	817,602	819,040	0.15	*	818,840	818,840	0.00	369.4	818,840	818,840	0.00	141.1
10-5-4-T	1,143,431	1,144,160	0.06	*	1,144,160	1,144,160	0.00	218.5	1,144,160	1,144,160	0.00	53.0
15-10-4-T	1,596,310	1,597,100	0.05	*	1,597,100	1,597,100	0.00	66.0	1,597,100	1,597,100	0.00	21.1
20-10-3-T	1,629,000	1,629,500	0.03	*	1,629,500	1,629,500	0.00	1,138.7	1,629,500	1,629,500	0.00	408.7
20-12-3-T	1,606,500	1,606,500	0.00	50.5	1,606,500	1,606,500	0.00	80.1	1,606,500	1,606,500	0.00	38.3
Average			**0.0461**				**0.0012**				**0.0000**	
25-12-3-L	1,677,200	1,677,400	0.01	*	1,677,234	1,677,400	0.01	*	1,677,233	1,677,400	0.01	*
25-12-3-T	2,046,933	2,047,000	0.00	*	2,046,933	2,047,000	0.00	*	2,046,933	2,047,000	0.00	*
12-5-4-L	1,240,248	1,245,160	0.39	*	1,243,262	1,245,160	0.15	*	1,243,019	1,245,160	0.17	*
12-5-4-T	1,398,454	1,404,640	0.41	*	1,402,329	1,404,520	0.14	*	1,403,550	1,405,580	0.05	*
30-12-3-L	2,016,178	2,016,600	0.02	*	2,016,178	2,016,600	0.02	*	2,016,245	2,016,600	0.02	*
30-12-3-T	2,491,406	2,492,500	0.04	*	2,491,490	2,492,600	0.04	*	2,491,437	2,492,600	0.04	*
20-12-4-L	1,943,545	1,943,800	0.01	*	1,943,800	1,943,800	0.00	778.0	1,943,800	1,943,800	0.00	635.2
20-12-4-T	3,113,085	3,113,170	0.00	*	3,113,170	3,113,170	0.00	75.1	3,113,170	3,113,170	0.00	40.8
15-8-4-L	1,495,243	1,497,100	0.12	*	1,497,000	1,497,000	0.00	3,228.6	1,497,000	1,497,000	0.00	4,309.9
15-8-4-T	1,655,075	1,656,040	0.06	*	1,656,040	1,656,040	0.00	4,356.3	1,656,040	1,656,040	0.00	792.3
25-12-4-L	2,439,300	2,440,500	0.05	*	2,439,810	2,440,500	0.03	*	2,439,585	2,440,800	0.04	*
25-12-4-T	3,705,997	3,707,790	0.04	*	3,707,390	3,707,390	0.00	578.0	3,707,390	3,707,390	0.00	6,483.4
30-15-4-L	2,918,409	2,918,800	0.01	*	2,918,510	2,918,600	0.00	*	2,918,464	2,918,800	0.00	*
30-15-4-T	3,274,228	3,278,860	0.12	*	3,275,538	3,278,480	0.08	*	3,275,004	3,278,280	0.10	*
40-15-3-L	2,689,060	2,689,200	0.01	*	2,689,060	2,689,200	0.01	*	2,689,060	2,689,200	0.01	*
40-15-3-T	3,329,567	3,330,500	0.02	*	3,329,567	3,330,500	0.02	*	3,329,567	3,330,300	0.02	*
Average			**0.0824**				**0.0314**				**0.0288**	
Optimal solutions			**16/50**				**37/50**				**39/50**	

Chapter 3

The multi-port continuous berth allocation problem with speed optimization

Bernardo Martin-Iradi[a], Dario Pacino[a], and Stefan Ropke[a]

[a]DTU Management, Technical University of Denmark,
Akademivej Building 358, 2800 Kgs. Lyngby, Denmark

Status: Published as a conference paper in *Lecture Notes in Computer Science*.

Abstract: We study the multi-port continuous berth allocation problem with speed optimization. This problem integrates vessel scheduling with berth allocation at multiple terminals in a collaborative setting. We propose a graph-based formulation and a branch-and-price method to solve the problem. The results show that the branch-and-price procedure outperforms the baseline solver. In our computational study, we highlight the trade-off between solution quality and computational complexity, as a function of the segment length used to model a continuous quay.

Keywords: Transportation, Maritime logistics, Container terminal, Exact methods

3.1 Introduction

The liner shipping industry is one of the major forms of international freight transportation. Seaborne trade and container throughput has continued growing steadily until 2019 and, despite the COVID-19 disruption in 2020, maritime trade recovered and is projected to expand by 4.3 % in 2021. The world fleet is also growing, not only in number of ships (more than 3 % in 2021), but also in size (the carrying capacity of mega-vessels rose from 6 to almost 40 per cent in the last 10 years) (UNCTAD, 2021). To accommodate the growing trade volume, ports and their container terminals need to either expand their capacity or improve the efficiency of their operations. Whereas the former usually requires a costly investment and in some cases it is not physically possible, the latter can be explored by means of operations research.

One of the key logistical operations in a container terminal is the berth allocation (Steenken et al., 2004). This operation is mathematically modeled as the berth allocation problem (BAP), where the aim is to assign berthing positions to incoming ships. The BAP is NP-hard (Lim, 1998) and has been extensively studied in the literature (Bierwirth and Meisel, 2015).

Most BAP studies consider either a discrete or a continuous quay. In the discrete variant, the quay is discretized into a set of berthing positions which can only serve one ship at a time. In the continuous version, ships can berth at any point within the quay as long as they respect a safety distance from other ships. The literature studies on the continuous BAP have approached the modeling part in different ways. Some studies use a continuous variable to define the berthing position of the ship (Kim and Moon, 2003; Lyu et al., 2022), whereas other studies divide the quay into segments of short length (e.g., 10 meters) and allow ships to occupy multiple consecutive segments based on their length (Imai et al., 2005; Meisel and Bierwirth, 2009). In practice, the position of the quayside bollards can restrict the berthing positions for the ships, strengthening the latter modeling approach. We observe that a solution to the discrete BAP is feasible for the continuous BAP but it is not guaranteed that a solution to the continuous BAP is feasible for the discrete version of the problem. As a result, the continuous BAP provides a better or equal solution than the continuous one, but it is normally harder to solve. Our study follows the second modeling approach and investigates this trade-off between solution quality and computational complexity.

The main cost driver for a liner shipping company (i.e., carrier) is fuel consumption. The relation between fuel consumption and the vessel's sailing speed is non-linear, which translates in the fuel consumption growing significantly at higher speeds. Therefore, optimizing the sailing speed is one of the main priorities for carriers. The mathematical problem that studies this operation is known as the vessel scheduling problem (VSP) and has been actively researched in the last two decades (Dulebenets et al., 2019). The VSP aims at defining the sailing speeds between consecutive ports (i.e., voyage leg) to optimize fuel consumption while guaranteeing a service frequency on the route.

The optimization of the BAP and VSP have helped significantly improve the efficiency of operations for terminal operators and carriers, but can potentially lead to logistics issues in practice. On one hand, berth allocation is myopic as most container terminals plan their berth independently from other terminals. This is motivated by the competitive environment and reticence to share information. As a result, if one of the terminals faces a congestion, delayed vessels can propagate the delay to the next ports in their routes, leading to higher operational costs for both carriers and terminals (Notteboom and Vernimmen, 2009). On the other hand, the VSP is mainly studied from the carriers' perspec-

tive, and does not explicitly account for the berth allocation at the terminals. Overseeing the berth assignments can potentially result in longer turnaround times for vessels.

Recently, efforts have been made to address these issues by exploring collaborative schemes that take advantage of information sharing. In Venturini et al. (2017) we see the first effort to integrate the BAP and VSP into the multi-port berth allocation problem with speed optimization (MBAP), which aims at planning the berth allocation of multiple terminals simultaneously while also optimizing the sailing speed of the ships. The MBAP relies in a high level of collaboration, and recent studies show that these types of collaborative problems can be mutually beneficial to both the carriers and terminal operations (Dulebenets, 2019; Martin-Iradi et al., 2022).

To the best of our knowledge, the MBAP has only been studied considering a strictly discrete quay. The contributions of this paper are three-fold: (i) we present two mathematical formulations for the MBAP with a continuous quay based on a graph representation of the problem, (ii) we define a new set of benchmark instances based on real port data, and (iii) we propose an efficient exact method for the problem and demonstrate its performance over state-of-the-art commercial solvers.

3.2 Problem formulation

In this section, we present two graph-based formulations for the MBAP with a continuous quay. We follow the modeling approach of diving the quay into segments of small size as in Meisel and Bierwirth (2009), where ships can occupy multiple segments simultaneously based on their length. Table 3.1 summarizes the notation of the problem.

It is general practice, especially on the discrete version of the BAP, to define a different handling time depending on the berthing position. For the study of a continuous quay, we follow the method presented in Meisel and Bierwirth (2009) where deviations from a preferred berthing position are penalized using a deviation factor $\beta \geq 0$ (relative increase in handling time per unit of distance, i.e., meters). Given the minimum handling time $h_0^{i,c}$ at the preferred berthing position, and the actual deviation from the chosen position Δb (measured in meters). The handling time at a given position b is computed as follows:

$$h_i^c(b) = (1 + \beta \Delta b) h_0^{i,c} \tag{3.1}$$

where $\Delta b = |b - x_0^{i,c}|$.

Table 3.1: Notation for the MIP formulation of the continuous MBAP

Sets and parameters:

N	Set of ships.
P	Set of ports.
T_p	Set of operational time instants at port $p \in P$.
S	Set of speeds.
L_p	Length of quay in port $p \in P$.
$P_i \subseteq P$	Set of ports planned to be visited by ship $i \in N$ sorted in visiting order.
$C_i. = \{1, ..., c\}$	Number of port calls for ship $i \in N$, one for each port visit.
$\rho(c)$	The port $p \in P$ corresponding to port visit number $c \in C_i$ for ship $i \in N$.
$\sigma(p)$	The port visit $c \in C_i$ corresponding to port $p \in P_i$ for ship $i \in N$.
$x_0^{i,c}$	The ideal berthing position for ship $i \in N$ at port visit $c \in C_i$ measured at the leftmost position of the ship.
$h_0^{i,c}$	Handling time at the ideal berthing position for ship $i \in N$ at port visit $c \in C_i$.
EST_i^c	The expected start time of berthing for ship $i \in N$ at port visit $c \in C_i$.
EFT_i^c	The expected finish time of berthing for ship $i \in N$ at port visit $c \in C_i$.
LFT_i^c	The latest finish time of berthing for ship $i \in N$ at port visit $c \in C_i$.
β	The relative increase in handling time per unit of distance.
$\Delta^{p,p'}$	Distance between ports $p, p' \in P$.
Θ_s	Travel time per unit of distance at speed $s \in S$.
Γ_s^i	Fuel consumption per unit of distance at speed $s \in S$ fro ship $i \in N$.
l_i	Length of ship $i \in N$.
F	Fuel cost in USD per tonne.
H	Cost of handling time in USD per hour.
D_i	Cost of delay time in USD per hour for ship $i \in N$.
I_i	Cost of waiting time in USD per hour for ship $i \in N$.

3.2.1 Network-flow formulation

Let $G = (O, A)$ be a directed graph formed by the sets of nodes O and arcs A. Additionally, we define the subset of arcs $A^k \subseteq A$ which denote the arcs available for a given ship $k \in N$.

We denote B_p to the set of quay segments of Φ meters for port $p \in P$. We define a node n for each port $p \in P$, berthing position $b \in B_p$, and time instant $t \in T_p$ Therefore, visiting a node can be interpreted as berthing at position b (left-most position) of port p at time instant t. Let $h_i^{c,b}$ be the handling time of ship i at port visit c and berthing position $b \in B_p$ and let $b_0^{i,c}$ be the berthing segment including position $x_0^{i,c}$.

$$h_i^{c,b} = (1 + \beta\Phi|b - b_0^{i,c}|)h_0^{i,c} \tag{3.2}$$

Equation 3.2 defines the computation of $h_i^{c,b}$ and it is adapted from Equation 3.1. Since each node refers to a single position b, we can pre-compute the handling time related to each node. The cost of a node $c_{p,b,t}^i$ for ship i is defined in Equation 3.3

$$c_{p,b,t}^i = H(h_i^{c,b}) + D_i(d_i^c) \tag{3.3}$$

where the delay d_i^c of ship i at port visit c is given as $d_i^c = \max(0, t+h_i^{c,b}-EFT_i^c)$.

Given the sequence of ports to visit by each ship, arcs are added to connect nodes of consecutive ports that correspond to feasible berths given the range of feasible sailing speeds. We add an arc between (p, b, t) and (p', b', t') for ship i if the ports are consecutive in the ships route $(p, p' \in P_i, p \prec p')$, and if the time difference allows to sail at a feasible speed $(t + h_i^{\sigma(p),b} + \Delta^{p,p'}\Theta_{MAX} \leq t')$ where Θ_{MAX} is the fastest feasible speed. Note that there is, at most, one arc between any pair of nodes. This arc corresponds to the speed level providing the lowest waiting time at the next port while still arriving on time. We do not allow the possibility of sailing faster to arrive to the same berthing time, as it does not provide any benefit and it only incurs in both higher waiting and fuel costs. Additionally, the time instants of the nodes need to satisfy the time windows $EST_i^c \geq t$ and $t' + h_i^{\sigma p',b'} \leq LFT_i^c$, and the left-most berthing position should consider the length of the ship $b + l_i^\Phi \leq B_p$ where l_i^Φ is the number of berthing segments that ship $i \in N$ occupies given a segment of length Φ. The cost c_a^i of an arc $a = ((p, b, t), (p', b', t'))$ for ship i is defined in Equation 3.4

$$c_a^i = I_i(t' - (t + h_i^{\sigma(p),b} + \Delta^{p,p'}\Theta_s)) + F(\Delta^{p,p'}\Gamma_s^i) \tag{3.4}$$

where $s \in S$ is the speed level associated with the arc.

Within the node set, we include $o, d \in O$ as artificial source and sink nodes respectively. Artificial source arcs are added for each ship connecting the source

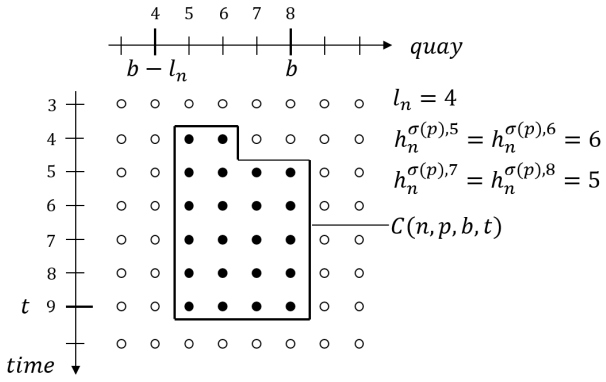


Figure 3.1: An example of the set $C(n, p, b, t)$.

node with all the nodes of the first port in the route. In the same way, we add artificial sink nodes for each ship connecting the nodes from the last port in the route with the sink node. Let $\delta_k^+(u)$ be the set of nodes that can be reached by following a single outgoing arc $a \in A^k$ from node $u \in O$ for ship $k \in N$. Likewise, let $\delta_k^-(u)$ be the set of nodes that can be reached by following a single incoming arc $a \in A^k$ from node $u \in O$ for ship $k \in N$. We use the notation $[x; y]$ to define an interval between x and y where y is included and $[x; y)$ where y is not.

For each ship $n \in N$, port $p \in P$, berthing position $b \in B_p$ and operating time instant $t \in T_p$, we define the set $C(n, p, b, t) \subseteq O$ that denote the graph nodes for ship n whose berthing period covers time t and whose berthing position covers segment b (i.e. nodes that are *in conflict* with any ship berthing at time t and position b).

An example is depicted in Figure 3.1 and the expression can be stated as follows:

$$C(n,p,b,t) := \left\{ v = (p, b', t') \in O \middle| b' \in \Big(\max(b - l_n^{\Phi}, 0); b \Big], t' \in \Big(\max(t - h_n^{\sigma(p),b'}, 0); t \Big] \right\}$$

Finally, let $x_{i,j}^k$ be a binary variable deciding if arc $(i, j) \in A^k$ is selected for ship $k \in N$ and let $c_{i,j}^k$ be the weight associated to the same arc. For simplicity, we add the cost of node i in the cost of the arc (i, j) to merge the node and arc costs $c_{i,j}^k = c_i^k + c_{i,j}^k$.

$$\min \sum_{k \in N} \sum_{(i,j) \in A^k} c_{i,j}^k x_{i,j}^k \tag{3.5}$$

$$\sum_{j \in \delta_k^+(o)} x_{o,j}^k = 1 \quad \forall k \in N \tag{3.6}$$

$$\sum_{i \in \delta_k^-(d)} x_{i,d}^k = 1 \quad \forall k \in N \tag{3.7}$$

$$\sum_{i \in \delta_k^-(j)} x_{i,j}^k - \sum_{i \in \delta_k^+(j)} x_{j,i}^k = 0 \quad \forall j \in O \backslash \{o,d\}, k \in N \tag{3.8}$$

$$\sum_{k \in N} \sum_{i \in C(k,p,b,t)} \sum_{j \in \delta_k^+(i)} x_{i,j}^k \leq 1 \quad \forall p \in P, b \in B_p, t \in T_p \tag{3.9}$$

$$x_{i,j}^k \in \{0,1\} \quad \forall (i,j) \in A, k \in N \tag{3.10}$$

The objective function (3.5) minimizes the cost of the selected arcs as in Martin-Iradi et al. (2022) which is the weighted sum of operational costs, namely, waiting time cost, handling time cost, delay time cost and fuel consumption cost. Constraints (3.6) and (3.7) ensure that, for each ship, only one arc leaves from the source node and arrives to the sink node respectively. Constraints (3.8) enforce flow conservation ensuring that for each node, except the source and sink ones, there are as many incoming as outgoing arcs. Constraints (3.9) avoid overlapping of ships in time and space by ensuring that each berthing position is occupied by at most one ship at each time instant. Finally, constraints (3.10) define the binary property of the variable.

3.2.2 Set partitioning formulation

Only the set of constraints (3.9) is not independent between ships. We, therefore, exploit the structure of the formulation and apply Dantzig-Wolfe decomposition (Dantzig and Wolfe, 1960) and transform the network-flow formulation into a set partitioning problem (SPP) formulation where constraint (3.9) is handled in the master problem and each variable (i.e., column) refers to a whole feasible schedule of a ship along its route. According to Jans (2010), the pure binary nature of the variables of the network flow formulation allows us to impose binary conditions on the variables of the new master problem.

$$\min \sum_{j \in \Omega} c_j \lambda_j \tag{3.11}$$

$$\sum_{j \in \Omega} A_j^i \lambda_j = 1 \quad \forall i \in N \tag{3.12}$$

$$\sum_{j \in \Omega} B_j^{p,b,t} \lambda_j \leqslant 1 \quad \forall p \in P, b \in B_p, t \in T_p \tag{3.13}$$

$$\lambda_j \in \{0,1\} \quad \forall j \in \Omega \tag{3.14}$$

The set of all columns is comprised in Ω and the decision variable λ_j is set to 1 if column $j \in \Omega$ is chosen as part of the solution and 0 otherwise. We denote c_j as the cost related to column $j \in \Omega$. In order to replicate the objective of the MIP formulation, this cost consists of the idleness, handling cost, delay and bunker consumption cost of the ship denoted by the column. Let A_j^i be a parameter that is equal to 1 if column $j \in \Omega$ corresponds to ship $i \in N$ and 0 otherwise. Likewise, let $Q_j^{p,b,t}$ be a parameter that is equal to 1 if the ship of column $j \in \Omega$ is occupying position $b \in B_p$ at time instant $t \in T_p$ at port $p \in P$ and 0 otherwise.

The objective function (3.11) minimizes the cost c_j of the columns which corresponds to the same weighted sum as the objective function of the network-flow formulation. Constraints (3.12) ensure that one column is selected for each ship. Constraints (3.13) guarantee that each berthing segment of each port is covered by at most one ship at any time instant. Finally, constraints (3.14) set the binary property of the decision variables.

3.3 Solution method

To solve (3.11)-(3.14), we present a *branch-and-price* method. We notice that the number of possible paths for each ship in the network is prohibitively large even for small size instances. Therefore, we opt for exploring delayed column generation methods. Notice that, by decoupling the pricing problem into N independent sub-problems, each of them results in a single shortest path problem. Given the directed and acyclic nature of the network, the problem can be solved using efficient label setting algorithms.

At each iteration, we solve the master problem with a restricted set of columns and obtain the dual solution. With that solution, we solve each of the pricing problems, and if a solution with a negative reduced cost is found, we add the corresponding new column to the master problem. We keep iterating until no more negative reduced costs are found.

3.3.1 Branching

Completing the column generation procedure gives us the optimal solution for the linear relaxation of the problem, which does not guarantee the solution to be integer. To achieve integer optimality, we need to embed column generation in a *branch-and-bound* procedure. This whole process is also known as *branch-and-price*, where column generation is performed at each node in the *branch-and-bound* tree.

A poor branching strategy can lead to exploring an excessively large or unbalanced tree and therefore, slow convergence. This is the case when branching on the path variables of the set partitioning formulation or arc variables from the network-flow formulation. Moreover, branching in these variables can impose additional restrictions in the pricing problem.

We propose branching on a set of nodes that aims to create a balanced partition. Given a fractional solution, we start by grouping the berthing times and positions per ship and port visit. For each ship and port visit, we compute the average and variance of their solution berthing times and positions. This results in $2 \cdot |N| \cdot |C_i|$ candidates, and we select the case with the highest variance to branch on. As an example, we assume that the case with the highest variance is the berthing position of ship A at visit B, with an average berthing position X. Then, our branching strategy enforces ship A to berth to the left or right of position X at visit B. This branching strategy guarantees at least one candidate and aims to provide a balanced branch-and-bound tree.

3.3.2 Computing bounds

For some of the largest instances, even solving the root node with column generation can be computationally expensive. If the time limit is reached and the column generation procedure has not converged, we can derive a valid lower bound. In our case, given the convexity constraints of the master problem that ensure the solution to contain one column per ship, our valid lower bound can be computed as indicated in Equation 3.15. Let z^* be the solution to the master problem at the last iteration and $\bar{c}_j$ the minimum reduced cost of pricing problem of ship $j \in N$, if negative, otherwise zero.

$$z^{LB} = z^* + \sum_{j \in N} \bar{c}_j \tag{3.15}$$

Once a percentage of the total time limit has reached and if the branch-and price procedure has not converged, we solve the original integer version of the problem with all the column generated in the branch-and-bound tree. This allows to obtain an initial upper bound or tighten the current one.

3.4 Results

In this section we perform a computational comparison between the proposed method applied to the set partitioning formulation, and a baseline commercial solver applied to the network-flow formulation.

Table 3.2: Parameter values used to generate the instance set.

Number of ships to optimize	Number of external ships per port	Segment length (m)
4-15	3-5	10, 20, 50, 100

3.4.1 Instance generation

The benchmark instances provided by Venturini et al. (2017); Martin-Iradi et al. (2022) are defined for the discrete case of the MBAP and are not rich enough to capture the aspects of the new variant of the problem. For this reason, we propose a new set of instances for the continuous MBAP.

We consider three different ship types: (i) feeder or small ($l_n :\leq 200$ meters), (ii) medium ($l_n : 201 - 300$ meters), and large ($l_n :> 300$ meters). Larger vessels have a larger dead-weight and load capacity which implies a higher fuel consumption in general. Therefore, we define a different speed-fuel consumption relation Γ_s^i for each ship type, as well a different minimum handling time $h_0^{i,c}$ at each of the port visits.

All instances consider 3 terminals located in 3 of the main ports in northern Europe: (i) Rotterdam APMT ($L_1 = 1600$ meters), (ii) Bremerhaven NTB ($L_2 = 1800$ meters), and (iii) Hamburg EGH ($L_3 = 2100$ meters).

The duration of the ships' time windows (i.e., $EST_i^c, EFT_i^c, LFT_i^c$) is based on historical port call data.

We define six different ship patterns based on real port data, each having a given type of ship and visiting either 2 or 3 ports in different orders. The set N of ships for a given instance is sampled from the ship patterns.

Parameters such as the ideal berthing position for the ships or the position and duration of external ships $\bar{N}$ are selected at random. We assume that the entire quay is available for berthing, unless an external ship is occupying it. The distance between ports is computed based on the actual sea distance of the maritime routes. Finally, we consider a set of 10 different speed levels, ranging uniformly between 17-21.5 knots.

To generate the entire set of instances, we use 3 parameter values: (i) number of ships to optimize, (ii) number of external ships per port, and (iii) the length of the quay segments. Table 3.2 indicates the values used for each parameter. For each combination of number of ships to optimize and number of external ships, we generate three instances, each with a different randomized seed. Then, for

each instance, we divide the quay into segments of different length, resulting in a final set comprising 432 instances.

3.4.2 Comparison of exact methods

We set an algorithm time limit of 1 hour. In the case of the branch-and-price method, we allocate 90 % of the time limit for the standard branch-and-price procedure and 10 % to solve the integer problem with all the columns generated. The model is entirely written in *Julia* (Bezanson et al., 2017) and using *CPLEX v. 12.10* as the solver on a single thread. It has been tested in a Xeon Gold 6226R with 64 GB of memory.

The results are summarized in Table 3.3, where we compare the branch-and-price method with the network-flow formulation solved by CPLEX. The instances have been grouped per number of ships to optimize, and the segment size. Each row comprises 9 instances. We compute the number of instances solved to optimality, the average computational time, and the optimality gap. We also compute the improvement in objective value with regards to the instances with a coarser segment length.

From the results in Table 3.3, we observe that the proposed branch-and-price method performs better than commercial solvers on the network-flow formulation. In the case of CPLEX, it is able to solve a few instances faster than the branch-and-price method, but we noticed that for instances with highly dense graphs (see Table 3.4), CPLEX runs out of memory when building the model. The average computational time in these cases corresponds to the runs of instances where CPLEX was not interrupted due to memory issues. On the contrary, branch-and-price finds a feasible upper bound for all instances within the time limit. We can observe that the impact in the solution quality of having a shorter segment length is significant. The operational costs can be reduced in more than 7 % in some cases with a segment of 10 meters instead of 100 meters. However, this improvement comes at the expense of higher computational needs. From a practical perspective, solving the problem with a more coarse segment length could be useful when quick solutions are needed, for instance, when facing a disruption and needing to re-plan or when testing multiple scenarios.

Table 3.5 shows the average results grouped by segment length. Dividing the quay in segments of 100 meters allows the method to solve all the instances, while for segments of 10 meters, we can solve 74 % of the instances maintaining a tight optimality gap. Moreover, halving the segment length from 100 to 50 meters helps saving more than 2 % in operational costs with an average run time of less than 5 minutes. Regarding individual operational costs, we observe that using shorter segments allows to achieve significant savings in waiting time and delay.

Table 3.3: Computational results for the set of instances grouped by number of ships to optimize and the length of the quay segments. "*" indicates that all instances reached the memory limit.

Instance		Branch-and-price				CPLEX		
Number of ships	Segment length (m)	Optimal instances (out of 9)	Optimality gap (%)	Time (s)	Objective improvement (%)	Optimal instances (out of 9)	Optimality gap (%)	Time (s)
4	10	9	0.00	114.0	-1.94	0	*	*
4	20	9	0.00	30.4	-1.63	6	0.00	512.5
4	50	9	0.00	19.7	-1.10	9	0.00	74.5
4	100	9	0.00	17.0	-	9	0.00	7.9
5	10	9	0.00	43.9	-1.98	0	*	*
5	20	9	0.00	24.1	-1.82	2	0.00	336.3
5	50	9	0.00	15.2	-0.93	9	0.00	119.3
5	100	9	0.00	13.8	-	9	0.00	13.2
6	10	9	0.00	76.3	-1.33	0	*	*
6	20	9	0.00	25.8	-1.13	4	0.00	1447.4
6	50	9	0.00	18.2	-0.83	9	0.00	128.7
6	100	9	0.00	16.3	-	9	0.00	13.1
7	10	9	0.00	282.4	-1.44	0	*	*
7	20	9	0.00	233.3	-1.29	0	*	*
7	50	9	0.00	80.9	-0.78	9	0.00	289.3
7	100	9	0.00	25.6	-	9	0.00	30.1
8	10	9	0.00	63.3	-2.06	0	*	*
8	20	9	0.00	20.6	-1.72	0	*	*
8	50	9	0.00	28.6	-0.57	9	0.00	311.4
8	100	9	0.00	17.7	-	9	0.00	32.8
9	10	7	0.21	994.0	-7.89	0	*	*
9	20	7	0.10	886.0	-7.46	0	*	*
9	50	8	0.00	571.2	-6.74	8	0.00	495.3
9	100	9	0.00	217.7	-	8	0.00	48.6
10	10	6	0.21	1160.8	-6.31	0	*	*
10	20	7	0.07	860.9	-5.94	0	*	*
10	50	9	0.00	137.6	-1.73	9	0.00	585.1
10	100	9	0.00	38.2	-	9	0.00	58.0
11	10	7	0.12	1225.3	-6.70	0	*	*
11	20	9	0.00	787.9	-6.26	0	*	*
11	50	9	0.00	186.1	-1.29	9	0.00	919.2
11	100	9	0.00	42.8	-	9	0.00	84.4
12	10	6	0.15	1480.8	-5.68	0	*	*
12	20	6	0.30	1189.8	-4.88	0	*	*
12	50	9	0.00	278.8	-4.13	8	0.04	1328.5
12	100	9	0.00	123.1	-	8	0.00	117.8
13	10	7	0.14	1131.9	-2.78	0	*	*
13	20	7	0.04	1150.2	-2.27	0	*	*
13	50	9	0.00	373.6	-1.41	6	0.18	1268.3
13	100	9	0.00	75.9	-	8	0.00	105.5
14	10	2	1.17	2974.9	-2.05	0	*	*
14	20	2	0.27	2595.2	-1.84	0	*	*
14	50	9	0.00	428.6	-1.22	6	0.00	1501.7
14	100	9	0.00	155.0	-	6	0.00	172.2
15	10	0	1.69	3365.4	-2.41	0	*	*
15	20	2	0.82	2606.4	-2.26	0	*	*
15	50	9	0.08	1115.1	-2.17	6	0.62	2064.8
15	100	9	0.00	534.9	-	6	0.13	583.9

Table 3.4: Average network sizes across instances with different quay segment lengths.

Segment length (m)	Average number of nodes	Average number of arcs
100	13,483	327,392
50	26,963	1,240,023
20	67,227	7,336,565
10	134,808	25,981,650

Table 3.5: Average operational cost savings compared to a segment length of 100 meters.

Segment length (m)	Optimal instances (%)	Optimality gap (%)	Time (s)	Waiting cost (%)	Delay cost (%)	Handling cost (%)	Fuel cost (%)	Total (%)
10	74.1	0.46	1076.1	-9.12	-12.46	-0.87	0.45	**-3.84**
20	78.7	0.19	867.6	-6.81	-10.29	-0.70	0.51	**-3.48**
50	99.1	0.01	271.1	-5.13	-8.93	-0.62	0.19	**-2.08**
100	100.0	-	106.5	-	-	-	-	-

The savings in handling time are lower but still positive. However, we notice that the fuel consumption costs increase marginally with shorter segments, suggesting that for most instances ships already sail at the slowest speed. Therefore, we can argue that, although using shorter segments does not necessarily translate into fuel savings, it helps to save in overall operational costs, by using the resources at the terminal more efficiently. As suggested in Martin-Iradi et al. (2022), the total savings arising from this collaborative problem could be distributed efficiently among the participating carriers and terminal operators, resulting in cost savings for all players and incentivizing further collaboration.

3.5 Conclusion

In this paper, we have studied a logistical problem that aims at simultaneously optimize the vessel scheduling and their berthing assignment in their port visits. We have modeled the continuous quay version of the problem as a network-flow formulation which we have re-formulated into a a set partitioning formulation. Decoupling the pricing problems per ship, allows to compute new columns by solving a shortest path problem. The results highlight the better performance of the proposed branch-and-price method compared to baseline solvers. Moreover, we show that using shorter quay segments can lead to a better use of the terminal resources and provide savings for carriers and terminal operators.

A natural next step for future work would be to study a different modeling approach for the continuous MBAP using a continuous variable to define the berthing position. Another aspect that deserves attention is the scalability of the method where exploring approaches to reduce the size of the graphs can help solving larger instances. Also, further research on possible valid inequalities could help fasten the algorithm. Moreover, we could explore ways to embed exact methods, such as the one presented, with heuristic procedures. This type of matheuristic could provide high quality solutions in shorter computational times.

References

Bezanson, J., Edelman, A., Karpinski, S., and Shah, V. B. (2017). Julia: A fresh approach to numerical computing. *Siam Review*, 59(1):65–98.

Bierwirth, C. and Meisel, F. (2015). A follow-up survey of berth allocation and quay crane scheduling problems in container terminals. *European Journal of Operational Research*, 244(3):12689, 675–689.

Dantzig, G. and Wolfe, P. (1960). Decomposition principle for linear-programs. *Operations Research*, 8(1):101–111.

Dulebenets, M. A. (2019). Minimizing the total liner shipping route service costs via application of an efficient collaborative agreement. *Ieee Transactions on Intelligent Transportation Systems*, 20(1):8315131.

Dulebenets, M. A., Pasha, J., Abioye, O. F., and Kavoosi, M. (2019). Vessel scheduling in liner shipping: a critical literature review and future research needs. *Flexible Services and Manufacturing Journal*, 33(1):43–106.

Imai, A., Sun, X., Nishimura, E., and Papadimitriou, S. (2005). Berth allocation in a container port: using a continuous location space approach. *Transportation Research Part B-methodological*, 39(3):199–221.

Jans, R. (2010). Classification of dantzig-wolfe reformulations for binary mixed integer programming problems. *European Journal of Operational Research*, 204(2):251–254.

Kim, K. H. and Moon, K. C. (2003). Berth scheduling by simulated annealing. *Transportation Research Part B: Methodological*, 37(6):541–560.

Lim, A. (1998). The berth planning problem. *Operations Research Letters*, 22(2-3):105–110.

Lyu, X., Negenborn, R. R., Shi, X., and Schulte, F. (2022). A collaborative berth planning approach for disruption recovery. *Ieee Open Journal of Intelligent Transportation Systems*, 3:153–164.

Martin-Iradi, B., Pacino, D., and Ropke, S. (2022). The multiport berth allocation problem with speed optimization: Exact methods and a cooperative game analysis. *Transportation Science*, 56(4):972–999.

Meisel, F. and Bierwirth, C. (2009). Heuristics for the integration of crane productivity in the berth allocation problem. *Transportation Research Part E: Logistics and Transportation Review*, 45(1):196–209.

Notteboom, T. E. and Vernimmen, B. (2009). The effect of high fuel costs on liner service configuration in container shipping. *Journal of Transport Geography*, 17(5):325–337.

Steenken, D., Voß, S., and Stahlbock, R. (2004). Container terminal operation and operations research - a classification and literature review. *Or Spectrum*, 26(1):3–49.

UNCTAD (2021). *Review of Maritime Transport 2021.* (Accessed on 07.05.2022).

Venturini, G., Iris, C., Kontovas, C. A., and Larsen, A. (2017). The multi-port berth allocation problem with speed optimization and emission considerations. *Transportation Research. Part D: Transport and Environment*, 54:142–159.

Chapter 4

An adaptive large neighborhood search heuristic for the multi-port continuous berth allocation problem

Bernardo Martin-Iradi[a], Dario Pacino[a], and Stefan Ropke[a]

[a]DTU Management, Technical University of Denmark, 2800 Kongens Lyngby, Denmark

Status: Under review at *European Journal of Operational Research*.

Abstract: In this paper, we study a problem that integrates the vessel scheduling problem with the berth allocation into a collaborative problem denoted as the multi-port continuous berth allocation problem (MCBAP). This problem optimizes the berth allocation of a set of ships simultaneously in multiple ports while also considering the sailing speed of ships between ports. Due to the highly combinatorial character of the problem, exact methods struggle to scale to large-size instances, which points to exploring heuristic methods. We present a mixed-integer problem formulation for the MCBAP and introduce an adaptive large neighborhood search (ALNS) algorithm enhanced with a local search procedure to solve it. The computational results highlight the method's suitability for larger instances by providing high-quality solutions in short computational times. Practical insights indicate that the carriers' and terminal operators' operational costs are impacted in different ways by fuel prices, external ships at port, and the modeling of a continuous quay.

Keywords: OR in maritime industry, Container terminal, Berth allocation problem, Speed Optimization, Heuristics

4.1 Introduction

The liner shipping industry is one of the major forms of international freight transportation. According to the report by (UNCTAD, 2020), seaborne trade and container throughput continued growing steadily until 2019. Despite the Covid disruption during 2020, maritime trade is projected to recover and expand by 4.3 % in 2021. The report also highlights that the world fleet is increasing,

not only in the number of ships (more than 3 % in 2021) but also in size. The share of the total capacity carried by mega-vessels increased from 6 % to 40 % in the last ten years.

This increase in demand, together with IMO's goal of reducing shipping emissions by 50 % by 2050 (IMO, 2018), requires container terminals to increase capacity and improve the efficiency and sustainability of their operations. The current growth of the vessel fleet and size directly impacts one of the most critical container terminal operations, namely the berth allocation (Steenken et al., 2004). Mathematically, this problem is denoted as the Berth Allocation Problem (BAP), which aims to assign incoming ships to berthing positions. The BAP can assume the quay to be discrete or continuous. In the discrete version, the quay is divided into positions where each can be occupied by one ship at a time. In the continuous BAP, ships can berth at any point in the quay while respecting a safe distance from other ships. Furthermore, the BAP can be dynamic or static. The static BAP assumes all the ships to be already at the port when the planning is done, whereas, in the dynamic version, ships can arrive at the port at different times during the planning period. It should be noted that the dynamic BAP is still a deterministic problem. The term *dynamic* refers to the different arrival times of each ship and not to the nature of the problem (Cordeau et al., 2005) like in, for example, vehicle routing problems. Figure 4.1 shows an example solution of the continuous and dynamic BAP.

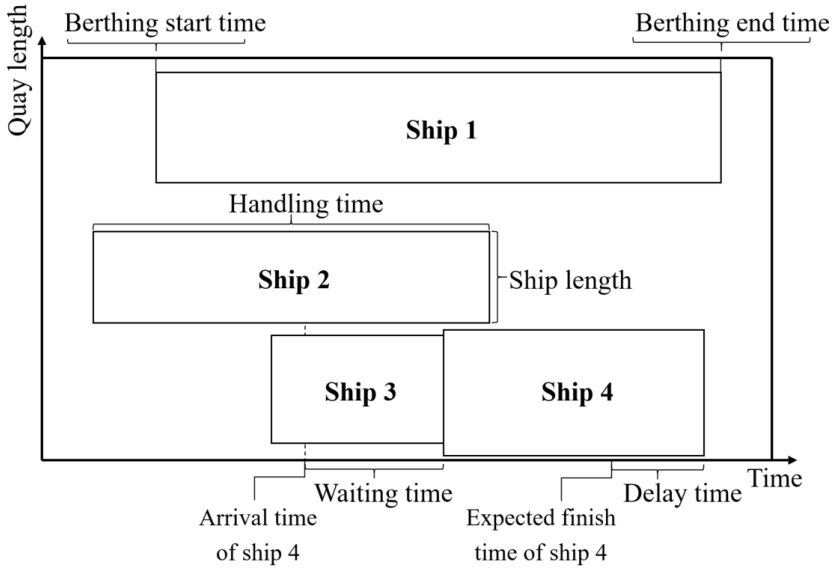


Figure 4.1: Example solution of the continuous and dynamic BAP for a port terminal with four vessels.

Terminals optimize their berth allocation to minimize their operational costs and the time ships need to spend at the port, including waiting time, handling time, and any delays. Due to the fierce competition between container terminals, they do not tend to share more information than is strictly required and do the planning independently from other terminals. One potential problem is that if congestion arises in a port, the affected ships can easily propagate delays to the following ports in their routes. One way to reduce the delay is for vessels to speed up when sailing between ports. However, sailing faster results in higher fuel consumption. This type of decision-making can be addressed by shipping line companies (i.e., carriers) in the Vessel Scheduling Problem (VSP). The goal of the VSP is to optimize the sailing speeds between consecutive ports in the vessel's route (i.e., voyage legs). Most VSP studies aim to minimize the vessels' fuel consumption, turnaround time at the port, and the number of vessels needed to ensure a given route frequency. However, the VSP has its limitations. One of them is the simplistic way of modeling the berthing times of ships at port. Whereas some studies model a simplified version of berth allocation, most do not include it. Not integrating the BAP into the VSP can lead to an unrealistic or even infeasible berth allocation and, as a result, delays that ships can propagate.

A problem that integrates the berth allocation with the vessels' speed optimization was first introduced by Venturini et al. (2017) as the Multi-port Berth Allocation Problem (MBAP). This problem selects a set of ships and a set of ports that are part of their routes and simultaneously optimizes the berth allocation at all the ports, together with the sailing time between ports. Venturini et al. (2017) studied the version of the problem with a discrete set of berthing positions. The problem involves the joint optimization of carrier and terminal operations and relies on the a priori agreement of the vessels and ports involved. Martin-Iradi et al. (2022b) showed that this type of collaboration could generate cost savings for the players involved (i.e., shipping carriers and terminal operators) but also benefit the environment as fuel emissions can be reduced significantly.

As mentioned earlier in this section, the main difference between the continuous and the discrete BAP is the flexibility in the berthing positions. The set of berthing positions in the discrete BAP corresponds to a subset of those from the continuous BAP. Therefore, one can argue that modeling the quay as continuous can lead to a more resource-efficient plan, as the optimal solution of the continuous BAP is equal to or better than that of the discrete BAP. However, this potential increase in solution quality comes at the expense of higher complexity, as the solution space becomes considerably larger.

Martin-Iradi et al. (2022a) studied the MBAP with a continuous quay (MCBAP) and highlighted the additional complexity, as the method proposed cannot scale to large instances. This scalability issue is addressed in our study, where we

employ heuristic methods that can tackle large real-world instances.

This paper makes the following four contributions:

1. We define a new mixed-integer problem (MIP) formulation for the MCBAP.
2. We present an instance generator for the MCBAP based on real-world port data, and define a set of benchmark instances that are made publically available.
3. We implement an Adaptive Large Neighborhood Search (ALNS) method tailored to the MCBAP and enhance it with a Local Search (LS) procedure based on ejection chains.
4. We show the viability of the ALNS method on real-size instances where it is able to find high-quality solutions faster than baseline commercial solvers.

The remainder of this paper is structured as follows. Section 4.2 comprises an extensive literature review of the MBAP together with other collaborative problems that include berth allocation or vessel scheduling. Section 4.3 describes the MCBAP in detail and presents the MIP formulation. The solution method is described in Section 4.4. Section 4.5 includes the instance generator's details and the computational study. The conclusions and further research is summarized in Section 4.6.

4.2 Literature review

One of the most important problems in a container terminal is the BAP, which has been studied extensively for over two decades. A survey of most of these studies is compiled in surveys by Carlo et al. (2014) and Bierwirth and Meisel (2015). Lim (1998) presented one of the first formulations of the problem and showed that it is NP-hard. Due to the additional hardness involving the BAP variant with a continuous quay, the use of heuristic methods has been predominant in the literature. The first studies of the continuous BAP were by Kim and Moon (2003) and Imai et al. (2005), where they presented MIP formulations to the problem and solved it using heuristic and meta-heuristic algorithms such as simulated annealing. Cordeau et al. (2005) covered both the discrete and continuous BAP and solved them using a taboo search. Guan and Cheung (2005) presented both a composite heuristic and a tree search exact method and showed that both outperformed commercial solvers. A hybrid variant between the continuous and discrete BAP was studied in Kordić et al. (2016), where ships can only berth in a subset of positions. Heuristic methods have been widely used when integrating the BAP with other terminal operations. One of

the main problems studied is the berth allocation and quay crane assignment problem (Iris and Lam, 2018). Iris et al. (2017) present a mixed integer problem formulation with additional enhancements and implement an ALNS heuristic to solve it, whereas Cheimanoff et al. (2022) uses a variable neighborhood search heuristic.

The VSP has also attracted significant attention in the literature. Dulebenets et al. (2019) present a comprehensive survey about the problem and highlight the potential of collaboration and information sharing as one of the future research directions. To the best of our knowledge, Fagerholt (2001) presented the first formulation of the VSP. Negotiating the port calls with the terminal operator Dulebenets (2018) indicates that carriers and terminal operators can achieve significant savings. A collaborative version of the VSP is presented by Dulebenets (2019), where terminal operators offer different port call durations and handling rates, leading to win-win situations. Fagerholt et al. (2010) aim at minimizing fuel consumption by optimizing the speed in a shipping route and modeling it as a shortest path problem. The authors discretize the possible arrival times at each port to approximate the non-linear relation between fuel consumption and sailing speed. Du et al. (2011) and Sun et al. (2018) integrate vessel speed optimization and berth allocation by considering ships within a certain sailing distance from the port.

In the last decade, together with the increased access to data, the study of problems that require collaboration between different stakeholders (e.g., carriers and terminal operators) has become more relevant. Wang et al. (2015) present two collaborative mechanisms that encourage sharing accurate information between carriers and terminal operators. Lalla-Ruiz et al. (2016) study the discrete BAP and present a cooperative search based on a grouping strategy where group members can only share information within the group. The collaborative berth allocation problem (CBAP) was introduced by Dulebenets et al. (2018) where a terminal planning its berth allocation can divert excessive demand to other terminals. Hellsten et al. (2020) present an ALNS heuristic for the port scheduling problem (PSP), where the aim is to schedule feeder vessels in multi-terminal ports. Collaboration has also been studied in disruption management. Lyu et al. (2022) present a formulation for re-planning the berth allocation and quay crane assignment and propose a heuristic method to solve it. Guo et al. (2022) study the berth assignment and allocation problem, which integrates the BAP with the berth assignment and line clustering problem. The first formulation of the MBAP was first introduced by Venturini et al. (2017). It solved a dynamic and discrete BAP in multiple ports while optimizing ships' sailing speed between ports. Martin-Iradi et al. (2022b) presented a branch-and-price method for the same problem and conducted a study of the collaboration mechanism using cooperative game theory. Martin-Iradi et al. (2022a) extended the branch-and-price method to the MBAP with a continuous quay, the same

problem of this study, and showed that exact methods are competitive for small and medium size instances but struggle to scale for larger instances. Recently, Yu et al. (2022) presented a genetic algorithm to solve a problem that integrates the BAP with speed optimization and vessel service differentiation to address both vertical and horizontal collaborations.

4.3 Problem description

The MCBAP integrates operational aspects concerning terminal operators and shipping carriers. We consider a set of ships and a set of terminals, each of them in a different port, to optimize their operations. Each ship visits all or a subset of the ports as a part of its route. The ships may visit the ports in different orders. The aim of the problem is to determine the berthing position and time of the ships at each of the terminals visited. Each terminal has a limited berthing space, given by the length of the quay. The service time required to load and unload the vessel is denoted as handling time and depends on the berthing position. We assume that it increases linearly with the deviation from an ideal position. Similar to most BAP studies, the berthing time and positions of ships are subject to a set of restrictions. Ships have a time window to be serviced also known as a port call, this is planned in advance and helps the operator to allocate berthing capacity and avoid excessive congestion. To allow for delays, the end of the time window is not strict but delays are penalized as they require the use of unexpected resources such as more worker hours.

It is well known that the relation between sailing speed and fuel consumption is non-linear. In fact, this relation is often approximated with a cubic function as in Equation (4.1) (Venturini et al., 2017; Martin-Iradi et al., 2022b)

$$F(s) = (\frac{s}{s_d})^3 F_d \tag{4.1}$$

where s is the sailing speed, s_d is the design speed of the ship, and F_d is the fuel consumption at the design speed. For our formulation, we discretize the set of possible sailing speeds and assume ships will sail the distance between ports at one of those speeds. Given the set of feasible sailing speeds, we can compute the corresponding set of fuel consumption rates. This assumption ensures a linear formulation of the problem.

Figure 4.2 shows an example graphical representation of the problem, highlighting the main operational aspects of a ship (i.e., ship 1). The ship berths strictly after its earliest start time but, due to the long handling time associated with the berthing position, the service time concludes after the expected finish time. The service time after the expected finish time is computed as a delay. After the ship is serviced, it can depart towards the next port. At the time of arrival,

the quay is occupied, and the ship needs to wait until ship number 3 finishes its berthing period. The time window this time is long enough to account for the waiting time, and the ship is able to finish without a delay.

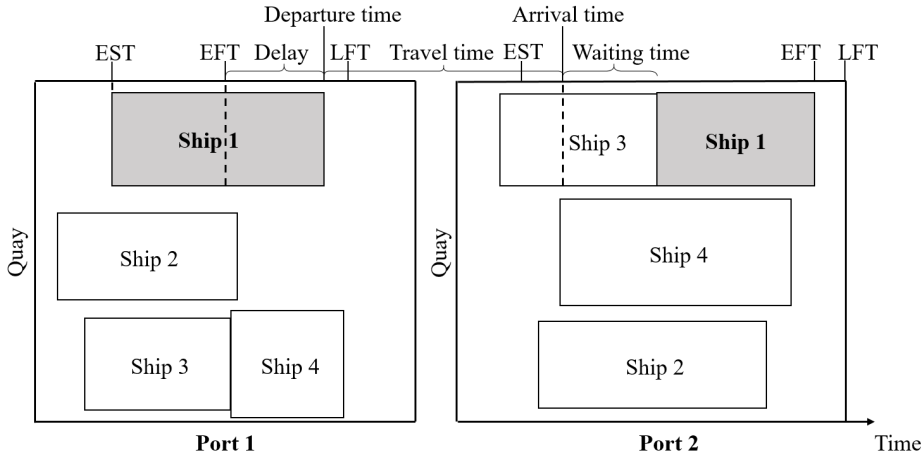


Figure 4.2: Example representation of a solution for the MCBAP with four ships visiting two terminals. The timeline of operations for ship 1 is defined in the top, where EST, EFT and LFT denote the earliest start time, the expected finish time, and the latest finish time of the ship at the port.

4.3.1 MIP formulation

We present a new MIP formulation for the MCBAP. This formulation is based on the one for the continuous BAP from Kim and Moon (2003) and the one for the discrete MBAP from Venturini et al. (2017):

Sets and parameters	
N	Set of all ships berthing at any of the ports.
$N^* \subseteq N$	Set of ships that we are optimizing.
$\bar{N} \subseteq N$	Set of external ships which are considered fixed.
P	Set of ports.
S	Set of speeds.
L_p	Length of quay in port $p \in P$.
$P_i \subseteq P$	Set of ports planned to be visited by ship $i \in N^*$ sorted in visiting order.
$C_i = \{1, ..., c_i\}$	Set of port calls for ship $i \in N$, one for each port visit. c_i is the last port visit, and the value is equal to the number of port calls.

ρ_i^c	The port $p \in P$ corresponding to port visit $c \in C_i$ for ship $i \in N$.
$N_p \subseteq N$	Set of ships that visit port $p \in P$.
$C_i^p \subseteq C_i$	Port call positions of ship $i \in N$ visiting port $p \in P$.
$x_0^{i,c}$	The ideal berthing position for ship $i \in N^*$ at port visit $c \in C_i$ measured at the leftmost position of the ship.
EST_i^c	The earliest start time of berthing for ship $i \in N^*$ at port visit $c \in C_i$.
EFT_i^c	The expected finish time of berthing for ship $i \in N^*$ at port visit $c \in C_i$.
LFT_i^c	The latest finish time of berthing for ship $i \in N^*$ at port visit $c \in C_i$.
β	The relative increase in handling time per unit of distance from the ideal berthing position.
$\Delta^{p,p'}$	Distance between ports $p, p' \in P$.
Θ_s	Travel time per unit of distance at speed $s \in S$.
Γ_s^i	Fuel consumption per unit of distance at speed $s \in S$ for ship $i \in N^*$.
l_i	Length of ship $i \in N$.
F	Fuel cost in USD per tonne.
H	Cost handling time in USD per hour.
D	Cost of delay time in USD per hour.
I	Cost of waiting time in USD per hour.
U	Cost penalty of exceeding the latest finish time in USD.

Decision variables	
$x_i^c \in \mathbb{R}^+$	the leftmost position of ship $i \in N$ at the quay for port visit $c \in C_i$.
$y_i^c \in \mathbb{R}^+$	the start time of berthing of ship $i \in N$ at port visit $c \in C_i$.
$v_{i,s}^c \in \mathbb{B}$	1 if speed $s \in S$ is chosen by ship $i \in N^*$ to sail between port visits c and $c+1$; $c \in C_i \backslash \{c_i\}$.
$d_i^c \in \mathbb{R}^+$	delay over EFT_i^c for ship $i \in N^*$ at port visit $c \in C_i$.
$u_i^c \in \mathbb{R}^+$	delay over LFT_i^c for ship $i \in N^*$ at port visit $c \in C_i$.
Auxiliary variables	
$\sigma_{i,j}^{c,c'} \in \mathbb{B}$	1 if ship i is positioned left of vessel j in the quay space at port visit $c \in C_i^p$ and port visit $c' \in C_j^p$ at port $p \in P$, 0 otherwise; $i, j \in N_p, i \neq j$.
$\delta_{i,j}^{c,c'} \in \mathbb{B}$	1 if ship i finishes berthing before vessel j starts berthing at port visit $c \in C_i^p$ and port visit $c' \in C_j^p$ at port $p \in P$, 0 otherwise; $i, j \in N_p, i \neq j$.
$r^{i,c} \in \mathbb{R}^+$	distance between ideal and actual berthing position of ship $i \in N^*$ at port visit $c \in C_i$.

Dependent variables	
$a_i^c \in \mathbb{R}^+$	arrival time of ship $i \in N^*$ at port visit $c \in C_i$.
$h_i^c \in \mathbb{R}^+$	handling time of ship $i \in N^*$ at port visit $c \in C_i$.

$$\min \sum_{i \in N^*} \Big(\sum_{c \in C_i} I(y_i^c - a_i^c) + H(h_i^c) + D(d_i^c) + U(u_i^c) + \sum_{c \in C_i \setminus \{c_i\}} F(v_i^c \Gamma_s^i \Delta^{\rho_i^c, \rho_i^{c+1}}) \Big) \tag{4.2}$$

$$x_i^c + l_i \leq L^p, \quad \forall i \in N_p, c \in C_i^p, p \in P \tag{4.3}$$

$$x_i^c + l_i \leq x_j^{c'} + L^p \left(1 - \sigma_{i,j}^{c,c'}\right) \quad \forall p \in P, i, j \in N_p, i \neq j, c \in C_i^p, c' \in C_j^p \tag{4.4}$$

$$y_i^c + h_i^c \leq y_j^{c'} + M \left(1 - \delta_{i,j}^{c,c'}\right) \quad \forall p \in P, i, j \in N_p, i \neq j, c \in C_i^p, c' \in C_j^p \tag{4.5}$$

$$\sigma_{i,j}^{c,c'} + \sigma_{i,j}^{c',c} + \delta_{i,j}^{c,c'} + \delta_{i,j}^{c',c} \geq 1 \quad \forall i, j \in N_p, i < j, c \in C_i^p, c' \in C_j^p, c < c', p \in P \tag{4.6}$$

$$y_i^c + h_i^c + \sum_{s \in S} v_{i,s}^c \Theta_s \Delta^{\rho_i^c, \rho_i^{c+1}} = a_i^{c+1} \quad \forall i \in N^*, c \in C_i \setminus \{c_i\} \tag{4.7}$$

$$a_i^c \leq y_i^c \quad \forall i \in N^*, c \in C_i \tag{4.8}$$

$$EST_i^c \leq y_i^c, \quad \forall i \in N^*, c \in C_i \tag{4.9}$$

$$y_i^c + h_i^c - EFT_i^c \leq d_i^c \quad \forall i \in N^*, c \in C_i \tag{4.10}$$

$$y_i^c + h_i^c - LFT_i^c \leq u_i^c \quad \forall i \in N^*, c \in C_i \tag{4.11}$$

$$\left(1 + \beta r^{i,c}\right) h_0^{i,c} = h_i^c, \quad \forall i \in N^*, c \in C_i \tag{4.12}$$

$$x_i^c - x_0^{i,c} \leq r^{i,c} \quad \forall i \in N^*, c \in C_i \tag{4.13}$$

$$x_0^{i,c} - x_i^c \leq r^{i,c} \quad \forall i \in N^*, c \in C_i \tag{4.14}$$

$$\sum_{s \in S} v_{i,s}^c = 1 \quad \forall i \in N^*, c \in C_i \setminus \{c_i\} \tag{4.15}$$

$$y_i^c, x_i^c \geq 0 \quad \forall i \in N, c \in C_i \tag{4.16}$$

$$a_i^c, h_i^c, d_i^c, u_i^c, r^{i,c} \geq 0 \quad \forall i \in N^*, c \in C_i \tag{4.17}$$

$$v_{i,s}^c \in \{0, 1\} \quad \forall i \in N^*, c \in C_i \setminus \{c_i\} \tag{4.18}$$

$$\sigma_{i,j}^{c,c'}, \delta_{i,j}^{c,c'} \in \{0, 1\} \quad \forall i, j \in N_p, i \neq j, c \in C_i^p, c' \in C_j^p, p \in P \tag{4.19}$$

The set of external ships $\bar{N}$ is considered fixed. Therefore, the corresponding set of decision variables $x_i^c, y_i^c, h_i^c, r^{i,c}$ for ships $i \in \bar{N}$ are constant and given as input to the problem.

The objective function (4.2) minimizes the operational costs of the carriers and terminal operators. This is measured as a weighted sum of the waiting time cost, handling time cost, delay cost, and fuel consumption cost. Constraints (4.3) ensure that each ship berths within the available space. Constraints (4.4) and (4.5) define the relative position of each pair of ships in each dimension by enabling the auxiliary variables $\sigma^{c,c'}$ and $\delta^{c,c'}$. The M value can be limited to the latest finish time of the pair of ships. Constraints (4.6) ensure that berthing periods do not overlap in time and space. Constraints (4.7) compute the arrival time to a port based on the sailing speed chosen to travel from the previous port. Constraints (4.8) and (4.9) enforce that the berthing starts strictly after arrival at port and after the time window starts, respectively. Constraints (4.10) compute the delay if the expected finish time is exceeded and constraints (4.11) define if the last finish time is respected. Constraints (4.12) compute the handling time for each ship and port visit while constraints (4.13) and (4.14) compute the deviation from the preferred berthing position. Finally. constraints (4.15) ensure that only one speed is chosen to sail between ports, and constraints (4.16) - (4.19) define the domain of the decision variables.

4.4 Solution method

To solve (4.2)-(4.19) we present an Adaptive Large Neighborhood Search (ALNS) algorithm. The ALNS algorithm, introduced by Ropke and Pisinger (2006), extends the large neighborhood search method by Shaw (1998). At each iteration, the method partially destroys and reconstructs a solution to generate a new solution. In our case, to destroy part of a solution, we remove the berthing time and locations of a subset of ships at a subset of ports. The combination of a scheduled berthing time and position for a ship at one of the ports in its route is denoted as a *port visit*, and we will refer to this term frequently in the remainder of the paper. Additionally, in some cases, we will refer to the scheduled port visit as a *rectangle*, in reference to how we can depict berthing position and time in a time-space diagram (e.g., see Figure 4.2).

4.4.1 Construction heuristic

The ALNS requires an initial solution to start with. We present a construction heuristic process for this step that aims at finding a good initial solution. Note that the BAP can be seen as a two-dimensional packing problem. However, in the continuous berth setting, the BAP has the increased complexity that

the length of the rectangles (i.e., port visits) vary depending on the berthing location in the quay. In the case of the MCBAP, we are solving multiple continuous BAP problems with the additional constraint that some of those berthing times depend on a sailing time. Moreover, the fact that ships follow different routes complicates the problem as greedy approaches become harder to apply. Our construction method prioritizes reducing the delay of ships at ports. We approach this by (i) trying to place port visits early in time and close to their ideal space, therefore reducing the handling time, and (ii) by reducing "useless" space, or, in other words, placing port visits efficiently not to create empty spots in the decision space that cannot be filled by remaining port visit. Notice that any possible solution is mathematically feasible since we allow it to exceed the latest finish time, and the time horizon is not limited. However, we aim to construct solutions where none of the ships exceed the LFT as those can be perceived as *infeasible* by the port operators and are also heavily penalized. The method acts as a greedy heuristic, where we schedule one port visit at a time. The port visit to schedule is selected as the *most constrained* one. To find it, we compute the set of feasible berthing positions and times for each ship and port visit. We consider a finite set of positions and times by dividing the quay into segments of a given length (e.g., 10 meters) and the planning horizon into intervals of 1 hour. For each time instant and segment, we compute if the ship can berth starting at that time and with its left-most side starting at the segment. We do not count berthing times exceeding the latest finish time to measure how constrained a ship's port visit is. From all unscheduled port visits, we define the one with the fewest possible positions as the *most constrained* one. We then schedule the port visit in one of the feasible positions. In fact, we do not consider the entire set but only the subset of feasible positions, where the port visit rectangle is directly adjacent to another scheduled port visit or to the limits of the decision space (i.e., the limit of the quay or planning horizon). From this subset of positions, we select the one resulting in the minimal change to the objective function. Besides the handling and delay cost directly computed when scheduling the port visit, we need to compute fuel consumption and waiting time costs. We consider these only if the previous port visit of the ship is scheduled.

Once a port visit is scheduled, we repeat the computation and selection of the *most constrained* unscheduled port visit and schedule it at the least costly *efficient* position. The procedure is described in Algorithm 4.1.

4.4.2 Removal and insertion operators

The goal of a removal operator is to select a set of scheduled port visits to be removed from the current solution. All the operators presented in this paper select K number of assignments to be removed, computed as a percentage ρ of the total number of port visits to be scheduled. It should be noted that

Algorithm 4.1 Construction heuristic

```
1: procedure constructionHeuristic(inst)(problem instance)
2:     unsch ← inst                ▷ initialize entire set of port visits to schedule
3:     sol ← ∅
4:     while unsch ≠ ∅ do
5:         unsch ← sort(unsch)     ▷ sort unplanned port visits by increasing
                                     number of feasible positions
6:         toSchedule ← popfirst(unsch)   ▷ get first port visit from the list
7:         planned ← false
8:         pos ← bestStartingPosition(toSchedule)  ▷ position at the earliest
                                     start time and closest to the ideal position
9:         while not planned do
10:            if feasible(pos, sol) then
11:                planned ← true
12:                sol ← plan(toSchedule, pos)        ▷ schedule the port visit
13:            else
14:                pos ← updatePosition(pos, sol)     ▷ update to next feasible
                                     position with lowest cost
15:            end if
16:        end while
17:    end while
18:    return sol
19: end procedure
```

removing port visits that are totally unrelated does not provide any potential gain. Therefore, a removal operator should aim at removing assignments that are related.

After applying a removal operator, the partial solution has K missing port visits that need to be scheduled. They need to be assigned efficiently while respecting the other assignments and ensuring that the solution remains feasible. This is the goal of the insertion operators.

4.4.2.1 Shaw removal

This operator, first introduced by Shaw (1998), selects the most related pairs of assignments. To select them, we define a measure of relatedness $M_{i,j}$ between assignments i and j in equation 4.20, similar to the one presented in Iris et al. (2017).

$$M_{i,j} = A|x_i - x_j| + B|y_i - y_j| + C|(y_i + h_i) - (y_j + h_j)|, \tag{4.20}$$

where x_i, y_i and $y_i + h_i$ are the berthing positions, berthing start time, and berthing end time of assignment i, respectively. A, B, and C are custom param-

eters that define the importance of each of the aspects. Observe that a lower value of $M_{i,j}$ translates into a higher level of relatedness. To select a total of K assignments, we select them following a greedy randomized criterion. To introduce randomness in the selection of the assignments, we define a parameter α. We sort all the port visit pairs in increasing order of $M_{i,j}$ and store them in the list Ω. We then select the i-th element of the list applying Equation (4.21):

$$i = \lceil |\Omega| \cdot p^{\alpha} \rceil, \tag{4.21}$$

where p is a random number $[0, 1)$. Note that if $\alpha = 1$, the selection is completely random, but as the value of α increases, the resulting value has a more deterministic behavior. The element selected will consist of two port visits to be removed. The selection process continues until K port visits are removed. Note that this method differs from the original method from Shaw (1998) in that the subsequent pairs do not necessarily need to be related with the first pair selected.

4.4.2.2 Time and space-relatedness removal

This removal uses a different relatedness measure. We first sort all port visits by cost. The cost B_i^c of port visit $c \in C_i$ for ship $i \in N^*$ is defined in Equation (4.22). It is measured by the ship's waiting, handling, and delay time at the port visit, plus half of the fueling costs from sailing from the previous port (if any) and to the next port (if any).

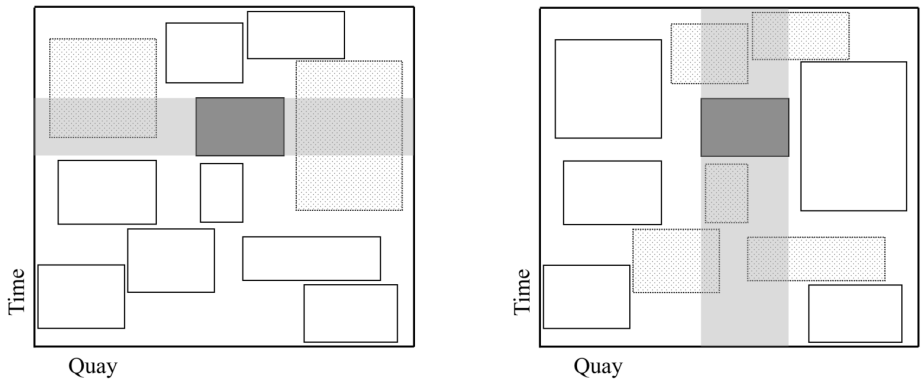
$$B_i^c = Hh_i^c + Dd_i^c + I(y_i^c - a_i^c) + \frac{F_i^c}{2} \tag{4.22}$$

F_i^c is the fuel costs associated with the previous and next port visits if any. For example, if the ship sails from a previous port visit c_p to port visit c, and then continues to the next port visit c_n, then the fuel costs are computed as in Equation (4.23).

$$F_i^c = F(v_i^{c_p,c} \Gamma_s^i \Delta^{\rho(c_p),\rho(c)}) + F(v_i^{c,c_n} \Gamma_s^i \Delta^{\rho(c),\rho(c_n)}) \tag{4.23}$$

In the case that port visit c is the first or last port visit in the route for the ship, the corresponding missing sailing leg is removed from the fuel cost computation.

We then select the i^{th} most expensive assignment applying Equation (4.21) and remove all *neighbor* assignments. We define as *neighbors* all the assignments that are within a *distance* of the assignment. We consider the *distance* in both time and space. If an assignment is depicted as a rectangle in a time-space diagram of the port, the *neighbor* area represents the one that overlaps in time or space with it. All other assignments that overlap partially or completely with the neighbor area are considered neighbors and removed. We then select the most expensive assignment and remove all neighbor assignments. We repeat the process until K assignments are removed.



(a) Time-related port visits (b) Space-related port visits

Figure 4.3: Neighbor port visits in time and space for a given port visit in dark grey.

Figure 4.3 shows an example of neighbor port visits in time and space. Depending on the dimension considered we define the two removal operators as *cost-time removal* and *cost-space removal*.

4.4.2.3 Random removal

We also consider a fully randomized destroy operator. It randomly selects K assignments to be removed. The goal of this operator is not to select relevant port visits to remove but rather to help diversify the search.

4.4.2.4 Randomized greedy insertion

This method follows the same procedure as the construction heuristic with the addition of a randomized component when selecting the port visit to schedule at each step.

All unplanned port visits are sorted based on the number of available insertion positions. An available insertion position is one that maintains a feasible solution. For instance, the port visit needs to ensure that the previous, or following port visits, are connected through a feasible sailing speed if any of these are already scheduled. We select the port visit using a randomization parameter γ in the same way that α is used in Equation 4.21. This prioritizes the port visits with fewer available insertion positions. The selected port visit is scheduled in the position that increases the objective function the least (i.e., lowest cost). The process iterates by recalculating the new number of insertion positions for the remaining port visits.

4.4.2.5 κ-regret insertion

This insertion method is based on the *regret-k* heuristic presented in Potvin and Rousseau (1993). This method has an additional *look-ahead* component compared to a basic greedy heuristic. For each of the port visits, we compute the κ best scheduling positions, and we then measure the *regret* cost for each of them as the difference between the best and κ-best positions. The one with the highest regret cost becomes the next port visit to plan. The process is described in Algorithm 4.2.

Algorithm 4.2 κ-regret insertion

1: **procedure** κREGRETINSERTION(sol, $unsch$, κ)(partially destroyed solution, set of port visits to schedule, and the parameter κ)
2: **while** $unsch \neq \emptyset$ **do**
3: $order \leftarrow \emptyset$ ▷ initialize empty list
4: **for all** $portVisit \in unsch$ **do**
5: $[pos] \leftarrow findBestPositions(\kappa)$ ▷ compute κ best insert positions
6: $regretCost \leftarrow c(pos[\kappa] - pos[1])$ ▷ compute regret cost
7: $order \leftarrow sortList(portVisit, regretCost)$ ▷ sort list by regret cost
8: **end for**
9: $sol \leftarrow plan(order[1])$ ▷ plan selected port visit
10: $unsch \leftarrow pop(order[1])$ ▷ update set of unplanned port visits
11: **end while**
12: **return** sol
13: **end procedure**

4.4.2.6 Packing greedy insertion

This insertion method is similar to the randomized greedy insertion described in Section 4.4.2.4. The main difference is the position where the port visits are planned. Scheduling the port visits in a position with lower objective value can lead to the creation of empty spaces and, therefore, to inefficient use of the decision space. This method restricts the set of possible insertion positions to the ones *strictly adjacent* to other scheduled ships, or to the limits of the quay or planning horizon. By *strictly adjacent*, we mean that the port visit to schedule needs to berth strictly next to another ship during at least one interval of time (e.g., one hour) or berth strictly before (or after) another ship with at least one quay segment in common. Also, we consider berthing positions where one of the sides is at one end of the quay, or if the berthing period starts or ends at the earliest and latest possible berthing time, respectively. Figure 4.4 shows some example positions considered.

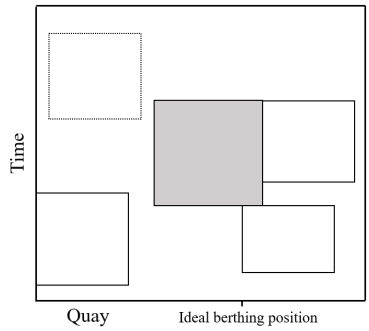


Figure 4.4: Graphical representation of example positions (continuous line) *strictly adjacent* to the grey ship or the quay space. The position represented with a dashed line is not part of the set of positions as it is not adjacent to another planned ship or the boundaries of the decision space.

4.4.2.7 Arrival greedy insertion

This method is identical to the one presented in Section 4.4.2.4 with the only difference that instead of sorting the unplanned port visits by increasing the number of feasible insertion positions, we sort the unplanned port visits by the earliest possible arrival time. One of the main goals of this method is to schedule port visits earlier, at the expense of a potentially higher cost, in order to increase the number of possible insertion positions for the remaining unplanned port visits.

4.4.3 Acceptance criterion

Once a new solution is reconstructed, we either accept it as the new current solution or reject it and reuse the previous one. We use a simulated annealing (SA) based criterion to take this decision. Such an acceptance criterion has been widely used for ALNS studies (see e.g. Ropke and Pisinger, 2006; Iris et al., 2017). We accept the new solution x' over the current one x if it is better ($f(x') < f(x)$), or if it is worse with a probability $e^{\frac{-(f(x')-f(x))}{T}}$, where T is the current temperature at a particular iteration, and $f(x)$ is the objective function. We define an starting an ending temperature, T_{start} and T_{end} respectively, and the cooling time t_{cool} that defines the duration of going from T_{start} to T_{end}. Based on these parameters, we can define the cooling factor τ ($0 < \tau < 1$), by isolating it from the formula $T_{end} = T_{start}\tau^{t_{cool}}$. This cooling factor allows us to compute the temperature at any given instant. Given temperature T at iteration i, we find the temperature T' to be used at iteration $i+1$ by computing

$T' = T\tau^{t_{it}}$, where t_{it} is the duration of the iteration i. Following the strategy used in Iris et al. (2017), we compute T_{start} and T_{end} based on the cost of the initial solution $f(x_0)$ described in Section 4.4.1, where ξ and ϕ define the percentage of the cost used to compute $T_{start} = \xi f(x_0)$ and $T_{end} = \phi f(x_0)$.

4.4.4 Adaptive weight adjustment

One of the main differences between the ALNS method and the standard Large Neighborhood Search (LNS) is the adaptive component of the former. The performance of the employed removal and insertion operators is measured at each iteration. These measures are then used to update the weight and, therefore, the probability of choosing the respective methods. The most common way of measuring the performance of a method is to give it a different score depending on the quality of the solution. In our case, we define four reward categories as shown in Table 4.1.

Category	Parameter
Current best solution	ψ_1
Better than current solution	ψ_2
Not better but accepted solution	ψ_3
Rejected solution	ψ_4

Table 4.1: Method reward categories

Let R and D denote the set of insertion and removal operators. Each removal and insertion method has a probability π_i^R, π_i^D respectively of being selected at each iteration. Throughout the algorithm run, the probability of selecting these methods gets updated depending on their performance. In our study, we update the probabilities after a Δ_{update} time interval. During these iterations we accumulate the sum of ψ_i^R, ψ_i^D rewards for each method, and update the weight ω_i^R, ω_i^D of each method as indicated in Equation 4.24

$$\omega_i^R = (1-\lambda)\omega_i^R + \lambda\psi_i^R, \quad \omega_i^D = (1-\lambda)\omega_i^D + \lambda\psi_i^D \tag{4.24}$$

where λ is a parameter between 0 and 1 that denotes the degree of adaptability of the method. If $\lambda = 0$, the weight remains equal to the previous one. This means that each method would have the same probability throughout the entire algorithm run, behaving like an LNS with multiple neighborhoods. If $\lambda = 1$, the new operator's probability solely depends on the score achieved during the last Δ_{update} and not on previous scores. It is common to use an intermediate value for λ strictly between 0 and 1. Once the weights are updated, the probability of each repair method π_i^R and destroy method π_i^D can be computed as indicated in Equation (4.25).

$$\pi_i^R = \frac{\omega_i}{\sum_{i\in R}\omega_i}, \quad \pi_i^D = \frac{\omega_i}{\sum_{i\in D}\omega_i} \tag{4.25}$$

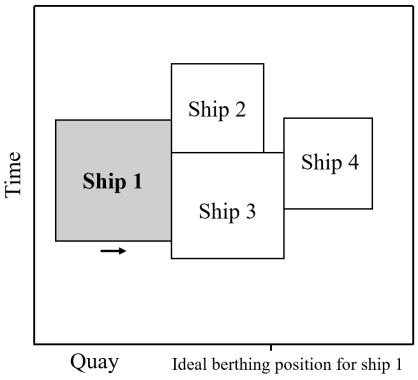
4.4.5 Local search

An extension of the method is implemented where we perform a local search procedure after reconstructing a new solution. This step aims to incrementally improve the solution by testing small adjustments to the port visits. The procedure is based on the *ejection chains* strategy used in many routing and network-based problems (see Glover (1992); Rego (1998); Bräysy (2003)). The idea, in our case, is to perturbate the solution by re-planning a port visit to a better position (i.e., lower operational cost) and iteratively re-plan any port visits that conflict with the change. The chain of perturbations is limited to a maximum number of port visits to re-plan K_{chain}, and it terminates if this limit is reached or if a conflict-free solution is achieved. Figure 4.5 shows an example of this move. Note that the handling time (i.e., the vertical dimension of the port visitsships) is reduced or increased for the ships as their position changes with respect to their ideal position. A pseudo-code of the procedure is described in Algorithm 4.3. The function $movePortVisit(p, n)$ performs the perturbation for a given port visit (i.e., ship n at port p). It should be noted that the direction is given by the first perturbation made. To find the direction of the first perturbation, we compute the cost variation of moving the port visit in three directions: (i) one segment length towards the ideal position along the spatial axis, and (ii) one time instant earlier and (iii) one time instant later along the temporal axis. The direction in the spatial axis is checked if the port visit is not scheduled already at its ideal position. Once the perturbation is performed in the chosen direction, the following port visits in conflict are perturbed in the same direction.

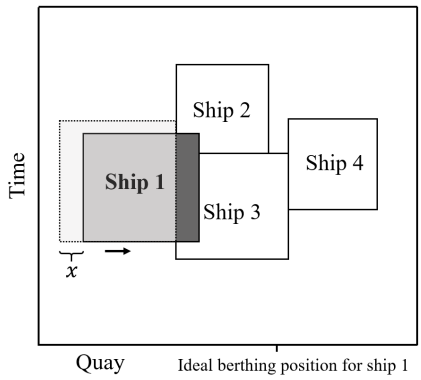
A high value of K_{chain} increases the probability of finding a better solution and the number of operations to compute. The parameter K_{chain} should leverage both solution quality and low computational complexity. Therefore, we define the value of K_{chain} to depend on the number of instances ships and equal to $K_{chain} = 2 \cdot |N|$. The reason for $K_{chain} > |N|$ is that for some movements, a conflicting port visit may require multiple perturbations to achieve a feasible new position, and selecting a lower K_{chain} value may be too restrictive.

4.4.6 Algorithm overview

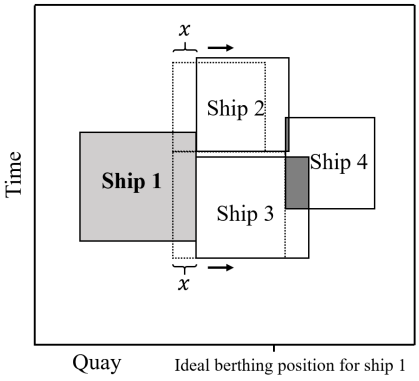
The overview of the solution method is summarized in Algorithm 4.4. Due to the additional computational effort of the local search procedure, we do not execute it at each iteration. Instead, we only perform it if the reconstructed solution is better than the current one. This reduces the number of times that the local search is performed, allowing the algorithm to perform more iterations while at the same time filtering the times the local search is performed to those where we already have promising solutions.



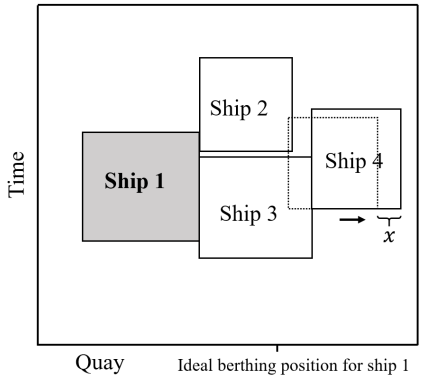
(a) The direction of move for ship 1 is defined.



(b) The move generates conflicts with ships 2 and 3.



(c) Moving ships 2 and 3 in the same direction generates a new conflict with ship 4.



(d) Moving ship 4 in the same direction ends the ejection chain by finding a new feasible solution.

Figure 4.5: Example representation of a local search step. The chain of moves originates from ship 1 being moved one step x towards its ideal berthing position. The port visit in grey depicts the first ship to move, and the dark grey indicates an overlapping area. The dashed rectangles represent the original position of the ship before the move.

Algorithm 4.3 Local search procedure

procedure LOCALSEARCH(sol, K_{chain})(current solution, and the length of the ejection chain (i.e., the maximum number of port visit moves))
  $done \leftarrow false$ ▷ initialize termination criterion
  while not $done$ **do**
    $nextSol \leftarrow sol$ ▷ initialize current best solution
    $\Delta \leftarrow 0$ ▷ initialize delta cost variation
    for all $p \in$P **do**
      for all $n \in N_p$ **do**
        $toMove = [(p, n)]$ ▷ track the port visits to re-plan
        $sol' \leftarrow sol$ ▷ initialize copy of current solution
        while $k \leq K_{chain}$ and $toMove \neq \emptyset$ **do**
          $(p, n) \leftarrow pop(toMove)$ ▷ get port visit to re-plan
          $sol' \leftarrow movePortVisit(p, n, sol')$ ▷ move port visit
          $toMove \leftarrow computeConflicts(sol')$ ▷ check for conflicts
          $k \leftarrow k + |toMove|$ ▷ update the ejection chain length
        end while
        if $toMove = \emptyset$ **then**
          $\delta \leftarrow computeDeltaCost(sol, sol')$ ▷ cost variation
          if $\delta < \Delta$ **then**
            $\Delta \leftarrow \delta$ ▷ update best delta cost
            $nextSol \leftarrow sol'$ ▷ update current best solution
          end if
        end if
      end for
    end for
    if $\Delta < 0$ **then**
      $sol \leftarrow nextSol$ ▷ update solution to return
    else
      $done \leftarrow true$ ▷ no improving neighbor solution
    end if
  end while
  return sol
end procedure

4.5 Computational results

In this section, we first describe the generation process for the set of benchmark instances, and we then perform a computational study where we cover both the performance of the method and practical insights of the problem.

Algorithm 4.4 Adaptive large neighborhood search procedure

procedure ALNS($inst, param$)(a problem instance and a parameter setting for the algorithm)
  $\psi, \pi \leftarrow initialize(inst)$ ▷ initialize operator selection parameters
  $sol \leftarrow constructHeuristic(inst)$ ▷ construct initial solution
  $bestSol \leftarrow sol$
  while timelimit not reached **do**
    $currSol \leftarrow sol$
    $removal, insertion \leftarrow selectOperator(\pi)$ ▷ select operators
    $sol \leftarrow insertion(removal(currSol))$ ▷ get new solution
    if $c(sol) < c(currSol)$ **then**
      $sol \leftarrow localSearch(sol)$
    end if
    if isAccepted(sol) **then**
      if $c(sol) < c(bestSol)$ **then**
        $bestSol \leftarrow sol$
      end if
      $currSol \leftarrow sol$
    end if
    $\pi \leftarrow updateOperatorParams(\psi)$
  end while
  return $bestSol$
end procedure

4.5.1 Instance generation

To the best of our knowledge, Martin-Iradi et al. (2022a) is the only study on the MCBAP. The instances presented in the study are rather small and limited. Therefore, we develop a more comprehensive set of benchmark instances. In the absence of real-life data, one could extend current benchmarks instances of the continuous BAP to multiple ports. Instead, we decided to use the public access to port data (Marine Traffic, 2023), and we validated it with additional data from an industrial research partner to create an instance generator for the MBAP with a continuous quay.

We consider three different ship types: (i) feeders or small vessels with a length of up to 200 meters, (ii) medium-size vessels with a length between 200 and 300 meters, and (iii) large vessels longer than 300 meters. Each ship type has a different speed-fuel consumption relation. Moreover, we consider three terminals at the three main ports in the north sea: (i) *Rotterdam APMT* (NLRTM), with a quay length of 1600 meters (APM Terminals, 2022b), (ii) *Bremerhaven NTB* (DEBRV), with a quay length of 1800 meters (APM Terminals, 2022a),

and *Hamburg EGH* (DEHAM), with a quay measuring 2100 meters (Eurogate, 2022). These three ports are relatively close to each other, and large, medium, and small vessels visit them in different sequences as part of their routes.

The duration of the vessel time window is based on the planned port call duration. The planned duration of a vessel's port call can often be updated the days previous to the arrival time. Therefore, we establish a fixed point in time for each ship two weeks before the actual arrival time and retrieve the planned port call duration as the time difference between the estimated time of arrival (ETA) and the estimated time of departure (ETD). We compute this by averaging the planned port call duration for each port and ship type berthing in a period of three months (January-March 2021). This value is also used to define the minimum handling time h_0 (see Table 4.2). Port service times that exceed 48, 72, and 96 hours for small, medium, and large ship types, respectively, are categorized as outliers and removed from the dataset. The reason for this is that such long service times usually involve maintenance or fueling operations that are not usually performed on a regular basis. Thus, they are not part of the problem.

Table 4.2: Minimum handling type in hours per ship type and terminal. These values define $h_0^{i,c}$.

Ship type \Terminal	**DEHAM**	**DEBRV**	**NLRTM**
Feeder	10.1	12.1	10.4
Medium	18.0	21.8	18.4
Large	41.0	33.7	26.7

We define six different ship patterns, each with a given route, type of ship, and length. All ships visit two or three ports in different orders. The N ships of an instance are sampled from the six patterns.

For each ship, we randomize the (i) desired berthing position at each port visited and (ii) the earliest start time EST_i^c, following parameters ensuring that feasible sailing times between ports exist. The estimated finish time EFT_i^c is computed by adding the corresponding value from Table 4.2 to EST_i^c ($EFT_i^c = EST_i^c + h_0^{i,c}$). In addition, we compute the latest finish time LFT_i^c by adding half of the difference between the minimum $h_0^{i,c}$ and maximum $h_{max}^{i,c}$ handling time that the ship can take at the port to EFT_i^c ($LFT_i^c = EFT_i^c + \frac{h_{max}^{i,c} - h_0^{i,c}}{2}$).

As input to the instance generator, we define the number of external ships at each port $Nout$ that are considered fixed. For each external ship, we randomly define: (i) the berthing position and time, (ii) the length (comprised between

180 and 330 meters), and (iii) the handling time, adapted to be proportional to the length of the vessel.

We assume the entire quay is available for berthing unless an external ship is occupying it. We also consider 10 different speed levels, ranging uniformly between 17-21.5 knots. The ALNS heuristic we present can handle any continuous value of the speed but not the MIP formulation we present. To ensure a fair comparison of the methods we employ a discretized set of speed levels in both the formulation and the solution method. Furthermore, the distance between ports is computed based on the actual sea distance of the routes.

4.5.1.1 Handling time

It is a general practice, especially on the discrete version of the BAP, to define a different handling time depending on the berthing position. For the continuous version implemented in this paper, we follow the handling time definition presented in Meisel and Bierwirth (2009) where deviations from a preferred berthing position are penalized using a deviation factor $\beta \geq 0$ (relative increase in handling time per unit of distance, i.e., meters). Given the minimum handling time $h_0^{i,c}$ at the preferred berthing position and the actual deviation from the chosen position Δb (measured in meters), the handling time is computed as follows:

$$h_i^c = (1 + \beta \Delta b) h_0^{i,c} \tag{4.26}$$

As a reference, the handling time ranges between 20 and 60 hours for medium vessels and between 30 and 110 hours for large ones. This is given by berthing at the best and worst places, respectively.

4.5.1.2 Time windows

In the MCBAP, there are two types of time windows:

- *The time window for each ship at each visited port.* This is given by the port call duration. The time window start must be respected, but the end can be exceeded. The berthing period can, therefore, exceed the end of the time window counting the additional time as a delay. We also consider a time window end that defines the latest finish time (LFT). Ideally, this time window must be respected. However, we allow violating this time window by adding a very high penalty cost.
- *The time window of the berthing positions.* This time window can be seen as the operational hours of a given berthing position or segment of the quay. We assume that all berthing positions are available at any time. It

Table 4.3: Parameter settings of the benchmark instance set.

Parameter	Seed	Number of ships	Number of external ships per port	Distance between positions
Values	1-10	30, 50, 70	5, 10	10, 20, 40, 80

should be noted that potential maintenance windows or partial closures of the quay can be modeled in the same way as an external ship occupying the given positions and time period.

4.5.1.3 Benchmark instances

Using the instance generator described in this section, we create a set of benchmark instances. The entire set comprises 240 instances. Each instance is a combination of the parameter values listed in Table 4.3: (i) a randomized seed, (ii) the number of ships to optimize, (iii) and the number of external fixed port visits per port, and (iv) the length of the quay segment used to define possible berthing positions.

4.5.2 Parameter tuning

The ALNS algorithm has a total of 18 algorithm parameters. These include parameters used to calibrate the operators, the selection and acceptance criteria, and the weight of the scores for new solutions (see Table 4.4). To select the best value setting for the algorithm parameters we conducted a parameter tuning. We selected a subset of 12 instances that are representative of the entire set. For the tuning, we run the automatic algorithm configurator *Pydgga* (Ansótegui et al., 2021) for the 30 generations with a time limit of 5 minutes for each ALNS run. The configurator allows to parallelize the process, and the entire parameter tuning lasted 8 hours. In Table 4.4, we define the domain of each algorithm parameter used for tuning and the found setting. The models and solution methods are written in *Julia* and run in a 2.90 GHz Intel Xeon Gold 6226R using one thread and 16 GB of RAM.

4.5.3 Method performance

To measure the quality of the method, we compare the presented method with its variants and with a baseline commercial solvers; *CPLEX v12.10*.

Table 4.5 compares the results between CPLEX and the ALNS method. We compute the objective gap for each instance run as $\frac{z_{obj}-z_{best}}{z_{best}}$ where z_{obj} is the objective value of the best solution of the run, and z_{best} is the best-known solution across all experiments. We have grouped all instances per number of ships,

Table 4.4: Studied range and chosen setting of algorithm parameters after the parameter tuning.

Symbol	Description	Min val	Max val	Tuned setting
ϵ	value used to compute the cooling ratio	0.005	0.2	**0.157**
φ	pct of initial solution obj used to define start temperature (when reheated)	0.01	0.05	**0.0246**
ξ	pct of initial solution obj used to define end temperature (to be reheated)	0.00005	0.001	**0.000269**
ρ	degree of destruction, pct of total port visits to be removed by the removal methods	0.3	0.6	**0.326**
A	weight for position deviation in shaw removal	0.5	2	**0.55**
B	weight for berthing start time deviation in shaw removal	0.5	2	**1.36**
C	weight for berthing end time deviation in shaw removal	0.5	2	**0.89**
α	randomness parameter for shaw removal method	1	3	**2.66**
γ	randomness parameter for random greedy repair method	1	3	**2.85**
μ	randomness parameter for arrival greedy repair method	1	3	**2.6**
κ	k-regret parameter	2	4	**2**
Δ	number of iterations between updating the weights of each method	0.01	0.05	**0.017**
η	parameter to adjust the importance of recent scores vs. previous weight	0.3	0.7	**0.456**
ψ_1	score when finding a current best solution	10	20	**11**
ψ_2	score when finding a solution better than the current solution	4	8	**4**
ψ_3	score when the solution is accepted	1	3	**2**
ψ_4	score when the solution is rejected	-	-	**0**
β	parameter that defines the position bounds for ship (times the length)	2	5	**4.02**

external ships, and distance between berthing positions. Each group contains 10 instances and is named *X-Y-Z* according to their common characteristics. X is the number of ships, Y is the number of external ships per port, and Z is the distance between consecutive positions considered in meters (i.e., segment length). Therefore, each row in the table corresponds to the average value across instances with different seed values. The ALNS is tested by running each instance 10 times and computing the average, best and worst run.

We observe that CPLEX scales poorly, especially in instances with more than 30 ships. The gap is better for the smallest instances but quickly worsens. On aver-

Table 4.5: Gap for the MIP formulation solved by CPLEX and the ALNS method with a time limit of 5 minutes and 1 hour. We also report the average gap between the best and worst runs of the ALNS. Each row corresponds to an instance group (i.e., the average results across 10 instances of the same size).

	5 minutes				1 hour			
Instance group	**MIP gap (%)**	**ALNS gap (%)**	**Best ALNS gap (%)**	**Worst ALNS gap (%)**	**MIP gap (%)**	**ALNS gap (%)**	**Best ALNS gap (%)**	**Worst ALNS gap (%)**
30_5_10	**10.1**	11.9	7.3	17.4	**1.5**	4.8	1.8	8.4
30_5_20	**7.4**	9.4	4.6	14.9	**1.7**	3.3	0.9	6.0
30_5_40	10.2	**6.9**	3.6	10.2	**1.2**	3.4	1.2	5.5
30_5_80	15.3	**7.5**	4.3	10.5	**2.1**	3.4	0.9	5.5
30_10_10	15.1	**10.0**	6.1	14.5	**2.7**	3.6	0.8	6.8
30_10_20	16.5	**8.1**	4.7	11.9	5.8	**2.9**	0.5	5.2
30_10_40	13.3	**5.9**	3.3	8.4	3.5	**2.3**	0.2	4.3
30_10_80	16.9	**5.2**	3.3	7.6	5.7	**2.3**	0.5	4.5
50_5_10	22.5	**14.4**	8.6	19.3	**2.2**	4.3	0.7	9.6
50_5_20	45.7	**11.4**	6.1	18.2	6.4	**3.6**	0.8	6.9
50_5_40	59.1	**8.8**	4.3	13.1	11.1	**2.9**	0.0	6.2
50_5_80	82.7	**7.2**	4.3	10.6	14.3	**2.9**	0.5	5.3
50_10_10	66.5	**10.3**	5.2	14.6	8.1	**2.9**	0.6	5.9
50_10_20	61.4	**8.3**	4.4	13.0	9.2	**2.2**	0.0	4.6
50_10_40	74.6	**7.0**	3.9	11.3	13.4	**2.7**	0.0	5.5
50_10_80	92.3	**6.2**	3.5	9.3	16.7	**2.2**	0.0	4.3
70_5_10	139.5	**13.8**	7.2	18.9	8.7	**3.9**	0.8	8.2
70_5_20	103.4	**10.9**	5.6	16.3	9.3	**2.4**	0.0	5.9
70_5_40	141.3	**7.8**	4.2	13.4	14.7	**2.4**	0.0	4.8
70_5_80	136.5	**5.8**	2.9	9.0	17.9	**2.3**	0.0	4.5
70_10_10	95.2	**11.7**	7.6	16.0	15.5	**2.7**	0.0	5.5
70_10_20	90.8	**8.8**	4.2	13.0	16.8	**2.3**	0.0	5.1
70_10_40	122.0	**7.3**	3.9	11.1	29.5	**2.3**	0.0	4.7
70_10_80	141.0	**5.5**	3.3	8.4	28.2	**2.1**	0.0	3.9
Average	65.8	**8.7**	4.8	13.0	10.3	**2.9**	0.4	5.7

age, the ALNS method outperforms the commercial solver by achieving tighter gaps in most instances. The gap is an indicator of the method performance relative to each other but does not provide an optimality guarantee. The lower bounds obtained with CPLEX indicate a high optimality gap. This could be due to a low-quality solution or a poor lower bound. Martin-Iradi et al. (2022b) indicated that the relaxation of the MIP formulation for the MBAP with a discrete quay could be poor and showed that a network-flow reformulation could tighten the relaxation significantly. As indicated in Martin-Iradi et al. (2022a), network-flow formulations for the MCBAP can suffer from scalability but show that the relaxation is stronger. We compare the results from the branch-and-price method presented in Martin-Iradi et al. (2022a) with the ALNS method. The formulation presented in Martin-Iradi et al. (2022a) is slightly different from the one addressed in this paper. The formulation from Martin-Iradi et al. (2022a) defines the latest finish time for each ship berthing at a port that must be satisfied. We have adapted the method from Martin-Iradi et al. (2022a) to

the formulation of this study. The branch-and-price method is based on a graph representation, and therefore, we need to establish the latest possible berthing time. This is set to 50 % more than the latest finish time. This allows the method to exceed the latest finish time while maintaining the graph at a reasonable size. This is a generous bound, and our empirical studies show that this bound does not affect the optimal solution. For that, we have also generated a new set of instances of similar size to the ones presented in Martin-Iradi et al. (2022a) using the instance generator defined in Section 4.5.1. The input parameters for the instance set are defined in Table 4.6.

Table 4.6: Parameter settings of the instance set based on the ones from Martin-Iradi et al. (2022a).

Parameter	**Seed**	**Number of ships**	**Number of external ships per port**	**Distance between positions (m)**
Values	1-5	4-15	3-5	10, 20, 40, 80

The entire set comprises 720 instances, one for each combination of input parameters. The results are shown in Table 4.7, where we have grouped the instances in batches of 60 according to the number of ships. We compare the branch-and-price method with the ALNS and MIP formulation presented in this study. For both the branch-and-price and CPLEX we compute their optimality gap, where we observe that the branch-and-price method achieves a better gap due to the tighter lower bound in most cases. We also compute the gap to the best-known solution for all three methods. In this case, we observe that CPLEX provides the best performance, showing that despite its poorer relaxation, the upper bounds found are near-optimal. The branch-and-price method still shows a robust performance but for short computational times and larger instances, the ALNS method is able to provide better solutions.

The ALNS method has two main components that differentiate it from other heuristics: (i) the adaptive procedure that guides the operator selection and (ii) the local search procedure that is performed when promising solutions are found. To quantify the impact of these two procedures, we compare the proposed method to its variants with and without each of the components. One variant is the method without its adaptive component (i.e., large neighborhood search (LNS)), meaning that each removal and insertion operator has an equal probability of being selected throughout the algorithm run. Another variant is the ALNS method without the local search (LS). The objective gap across the methods is compared in Table 4.8, with a time limit of 5 minutes, and 1 hour. For the short time limit, we see that the ALNS without the local search performs the best in most instances. Once the time limit is increased, the value of the local search is more apparent, and it provides the best results in most

Table 4.7: Performance comparison across 720 instances between the MIP formulation solved by CPLEX, the adapted branch-and-price method from Martin-Iradi et al. (2022a), and the ALNS method, with a time limit of 5 minutes and 1 hour. Each row shows the average gap values across all instances with same number of ships.

Number of ships	CPLEX Opt. gap (%)		Branch-and-price Opt. gap (%)		CPLEX Gap (%)		Branch-and-price Gap (%)		ALNS+LS Gap (%)	
	5 min.	1 hour	5 min.	1 hour	5 min.	1 hour	5 min.	1 hour	5 min.	1 hour
4	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	0.1	**0.0**
5	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
6	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	0.2	0.1
7	**0.0**	**0.0**	0.2	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	0.3	0.2
8	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	0.3	0.2
9	**0.0**	**0.0**	0.3	0.1	**0.0**	**0.0**	0.2	**0.0**	0.3	0.2
10	**0.0**	**0.0**	0.1	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	0.5	0.3
11	1.0	0.2	**0.3**	**0.1**	**0.1**	**0.0**	**0.1**	**0.0**	0.3	0.2
12	1.1	0.6	**0.8**	**0.2**	**0.1**	**0.0**	0.4	**0.0**	0.2	0.1
13	4.0	2.7	**1.1**	**0.6**	**0.1**	**0.1**	0.4	0.2	0.4	0.2
14	3.5	2.0	**1.2**	**0.4**	**0.1**	**0.1**	0.6	**0.1**	0.7	0.4
15	7.3	4.8	**1.7**	**1.0**	**0.3**	**0.1**	0.7	0.2	0.5	0.3
Average	1.41	0.86	**0.49**	**0.20**	**0.06**	**0.02**	0.20	0.05	0.30	0.19

instances.

To measure the impact of the local search procedure, we compare different strategies that differ in the frequency of execution of the local search procedure. We test three other variants of the algorithm in which the local search is called every 1, 2, and 4 iterations. The results are summarized in Table 4.9 where we can observe the increased computational complexity that the local search can add to each iteration. Performing the local search procedure very frequently can lead to good solutions in fewer iterations, but it also results in longer computational times. The proposed strategy performs the local search at iterations where the reconstructed solution is better than the incumbent one, and we show that this strategy performs the best. This method allows us to perform the local search in a fewer number of iterations, but at the same time, has the potential to result in promising and better solutions.

Tables 4.10 and 4.11 summarized the performance of the different removal and insertion operators used within the ALNS method. For each removal operator, we compute three metrics: (i) the percentage of iterations in which the operator was selected *IT*, (ii) the percentage of current best solutions found using the operator *NB*, and (iii) the percentage of times that the resulting solution was better than the current one using the operator *NC*. For the insertion methods, we also display a fourth column *T*, which indicates the percentage of time each operator has consumed from the total time spent repairing solutions. The time spent in removal methods is significantly lower than in insertion operators;

Table 4.8: Performance comparison between variants of the proposed ALNS method.

	5 minutes				1 hour			
Instance	ALNS + LS	ALNS (no LS)	LNS + LS	LNS (no LS)	ALNS + LS	ALNS (no LS)	LNS + LS	LNS (no LS)
group	gap (%)	gap (%)	gap (%)	gap (%)	gap (%)	gap (%)	gap (%)	gap (%)
30_5_10	11.9	**11.8**	12.0	13.6	**4.8**	4.9	5.5	5.9
30_5_20	9.4	9.6	**8.8**	10.6	**3.3**	3.6	3.4	4.0
30_5_40	**6.9**	7.3	7.3	7.8	3.4	3.3	**2.1**	2.5
30_5_80	7.5	7.5	**6.4**	7.0	3.4	3.7	**2.1**	2.3
30_10_10	**10.0**	10.9	10.6	11.8	**3.6**	3.7	4.2	4.7
30_10_20	**8.1**	8.2	8.6	9.6	2.9	**2.8**	3.1	3.3
30_10_40	5.9	6.3	**5.6**	6.3	2.3	2.4	**1.4**	1.7
30_10_80	5.2	5.4	**4.5**	5.3	2.3	2.3	**0.8**	1.0
50_5_10	**14.4**	15.5	15.1	18.2	**4.3**	5.1	7.1	9.4
50_5_20	11.4	**12.2**	12.8	16.3	**3.6**	4.4	5.1	7.7
50_5_40	**8.8**	10.2	11.0	13.2	**2.9**	3.9	3.8	6.2
50_5_80	**7.2**	8.3	8.2	10.6	**2.9**	3.0	3.4	4.6
50_10_10	**10.3**	11.4	12.0	14.4	**2.9**	3.9	5.1	7.1
50_10_20	**8.3**	9.2	9.9	12.5	**2.2**	3.3	3.6	5.7
50_10_40	**7.0**	8.1	8.1	10.9	**2.7**	3.5	3.1	4.9
50_10_80	**6.2**	6.6	6.6	8.7	**2.2**	2.6	2.6	3.1
70_5_10	**13.8**	14.7	14.6	17.4	**3.9**	5.4	6.7	10.9
70_5_20	**10.9**	11.4	11.4	14.4	**2.4**	4.6	5.1	8.5
70_5_40	**7.8**	10.1	9.4	13.1	**2.4**	3.9	3.9	7.1
70_5_80	**5.8**	7.8	7.6	11.0	**2.3**	3.5	3.2	5.5
70_10_10	**11.7**	12.6	11.8	14.8	**2.7**	4.3	5.2	8.3
70_10_20	**8.8**	10.3	9.6	13.3	**2.3**	4.2	4.6	7.6
70_10_40	**7.3**	9.2	8.4	11.9	**2.3**	3.8	3.9	6.2
70_10_80	**5.5**	6.8	6.6	9.4	**2.1**	3.0	3.0	4.5
Average	**8.7**	9.7	9.4	11.8	**2.9**	3.7	3.8	5.5

therefore, we do not compute this metric for the removal operators. We observe that the random removal is the better-performing removal method when the number of ships is 30. However, for larger instances, the cost-time removal performs better. This suggests that when the number of port visits increases, the probability of removing port visits that are not related at all also increases, making pure random methods less efficient. Furthermore, removing port visits that overlap in time is more effective than removing the ones that overlap in berthing space. While ships can berth at multiple positions along the quay without major delays and disruptions in their schedule, berthing earlier or later can negatively impact the sailing in the rest of the voyage legs. Therefore, their flexibility comes at a larger cost. We also notice that the Shaw removal becomes less useful in larger instances. This operator removes pairs of port visits at a time. These pairs can be highly unrelated between them, worsening the effects of the overall operator. Similarly to the random removal, this inter-relatedness issue increases with the number of port visits.

Regarding the repair methods, the κ-regret insertion method shows the best and more robust performance. Even if, in some cases, it is not the most frequently

Table 4.9: Algorithm comparison with different frequencies for the local search procedure with a time limit of one hour.

	LS every iteration		**LS every 2 iterations**		**LS every 4 iterations**		**LS when the solution is better than the incumbent**		
Instance group	**Gap (%)**	**Iter. x1000**	**Gap (%)**	**Iter. x1000**	**Gap (%)**	**Iter. x1000**	**Gap (%)**	**Iter. x1000**	**% of iter. with LS**
30_5_10	5.3	5.2	5.3	7.3	5.3	7.8	4.8	12.7	1.3
30_5_20	4.2	8.0	4.1	11.1	4.2	12.8	3.3	18.6	1.5
30_5_40	3.8	11.0	3.4	16.5	3.6	21.1	3.4	27.4	2.5
30_5_80	4.3	15.7	3.9	25.7	3.8	36.0	3.4	60.4	2.7
30_10_10	3.9	5.6	4.0	7.1	4.0	7.8	3.6	10.9	1.6
30_10_20	3.1	8.8	3.1	12.3	3.5	13.3	2.9	20.0	1.7
30_10_40	2.5	12.4	2.3	18.2	2.2	23.9	2.3	30.4	2.9
30_10_80	2.7	17.6	2.5	28.9	2.5	39.4	2.3	63.4	3.0
50_5_10	3.9	2.0	4.4	3.2	4.9	3.8	4.3	8.5	0.4
50_5_20	4.2	2.6	4.5	4.3	4.6	6.2	3.6	16.2	0.4
50_5_40	3.4	3.4	2.9	5.3	3.4	7.8	2.9	13.8	0.8
50_5_80	4.1	4.5	3.9	7.6	3.7	11.6	2.9	24.8	1.1
50_10_10	2.9	2.3	2.9	3.5	3.7	4.6	2.9	8.8	0.7
50_10_20	2.6	3.1	2.8	5.1	2.9	6.3	2.2	15.4	0.6
50_10_40	3.1	4.2	3.1	6.4	3.1	9.1	2.7	14.3	1.2
50_10_80	2.9	5.5	2.8	9.2	3.0	13.4	2.2	26.8	1.6
70_5_10	3.4	1.2	3.7	1.9	4.6	2.5	3.9	5.0	0.5
70_5_20	2.9	1.5	3.4	2.5	4.0	3.3	2.4	10.0	0.5
70_5_40	3.4	1.7	3.5	2.9	3.6	4.6	2.4	11.4	0.5
70_5_80	3.7	2.0	3.5	3.7	3.5	6.0	2.3	17.7	0.7
70_10_10	2.8	1.3	3.1	2.1	3.6	2.6	2.7	5.0	0.8
70_10_20	2.5	1.8	2.7	2.8	3.1	4.0	2.3	9.2	0.7
70_10_40	3.3	2.1	3.2	3.5	3.5	5.2	2.3	10.0	0.8
70_10_80	3.5	2.3	3.5	4.1	3.1	6.9	2.1	18.9	0.9
Average	**3.4**	**5.2**	**3.4**	**8.1**	**3.6**	**10.8**	**2.9**	**19.1**	**1.2**

Table 4.10: Performance summary of the four removal operators. The instances are grouped per number of ships.

Number of ships	**Cost-berth removal**			**Cost-time removal**			**Shaw removal**			**Random removal**		
	IT	NB	NC	IT	NB	NC	IT	NB	NC	IT	NB	NC
30	6	4	2	17	21	13	28	20	24	50	55	62
50	6	4	1	47	50	38	12	8	8	35	38	52
70	6	4	2	69	81	84	6	2	1	19	12	13

used operator, it is clearly computationally intensive, and more than half of the time is spent computing the insertions related to this method. The randomized greedy insertion is also relevant and becomes more effective in larger instances. The remaining two insertion operators, packing and arrival greedy, show a modest performance. Still, they also proved to help achieve some of the best-found

Table 4.11: Performance summary of the four insertion operators. The instances are grouped per number of ships.

Number of ships	Efficient packing insertion				Random greedy insertion				Arrival greedy insertion				κ-regret insertion			
	I	T	NB	NC	I	T	NB	NC	I	T	NB	NC	I	T	NB	NC
30	5	2	4	1	17	10	24	7	10	5	13	3	68	83	60	89
50	5	3	3	1	37	33	34	22	8	6	6	2	49	58	57	75
70	6	3	3	1	47	39	51	46	9	6	6	3	38	52	39	50

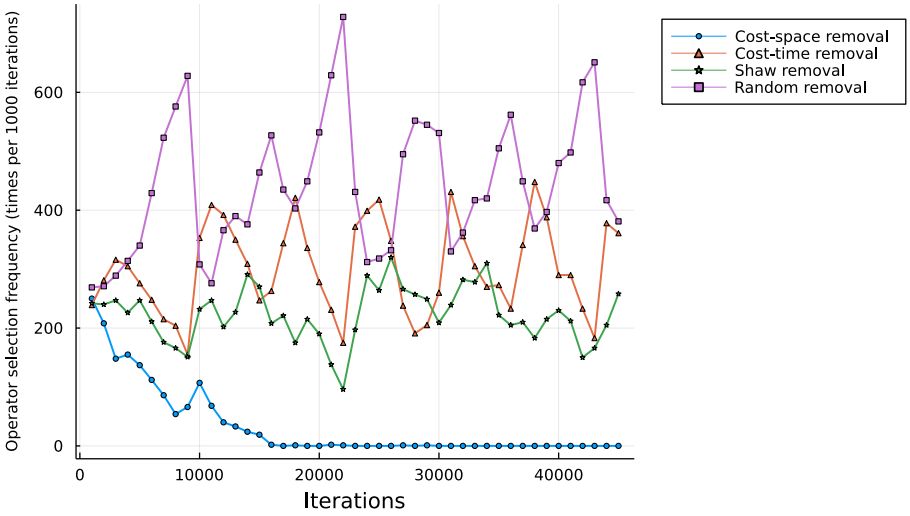


Figure 4.6: Usage of each removal operator during an algorithm run of 1 hour for an instance from group 30_10_10.

solutions during the algorithm run.

To better understand the algorithm's behavior and when the operators are used, we tracked the use of each operator during the algorithm run. Figures 4.6 and 4.7 show an example run of one hour for an instance with 30 ships, 10 external ships per port, and a quay segment length of 10 meters. We observe that the cost-berth removal is only used at the beginning of the run when each operator has a more balanced probability of being chosen. The packing greedy heuristic shows a similar behavior and correlates with the low usage of these two operators as shown in Table 4.10 and 4.11. Nonetheless, the rest of the operators are used for most algorithm runs. Some operators show an oscillating behavior, such as the cost-time removal operator. This behavior correlates with the temperature of the acceptance criterion, which is reheated periodically and suggests that the operator has a higher probability of being selected when the temperature is

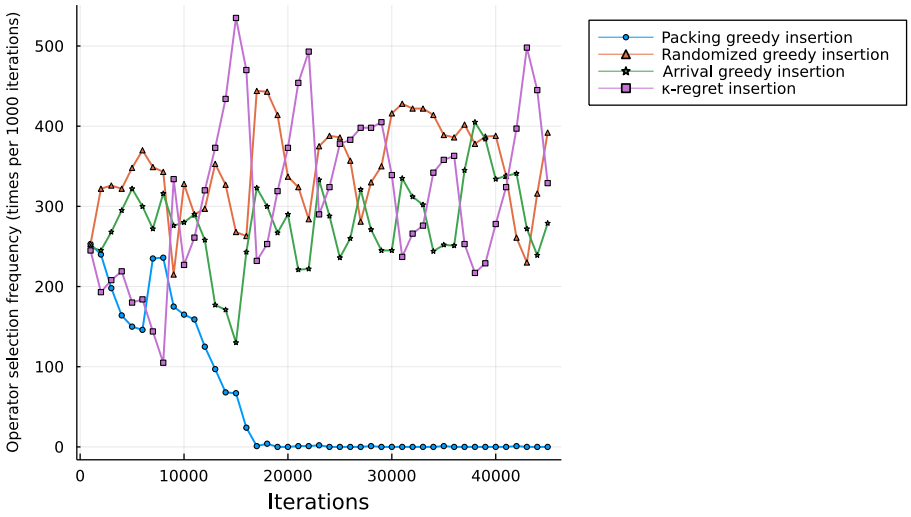


Figure 4.7: Usage of each insertion operator during an algorithm run of 1 hour for an instance from group 30_10_10.

higher.

Additionally, for both removal and insertion operators, we tested the algorithm removing the worst performing operators, one at a time, but the performance of the method worsened in all cases, indicating that all operators are to some extent useful and combine well together.

4.5.4 Practical impact

This problem involves two main stakeholders, namely, the terminal operators and the shipping carriers. The objective function covers the operational costs of both of them. This section disaggregates and analyzes the different operational costs by performing various sensitivity analyses.

In Table 4.12, we group the instances per quay segment length. We observe a natural trade-off here. A shorter segment length allows a more granular set of berthing positions and, therefore, a potentially better solution quality. However, this increases the complexity of the problem, and in the case of our method, it results in fewer iterations per hour. Despite performing less than a quarter of the iterations of the instances with 80 meters segments, the method finds better solutions for the shorter-segment instances. The improvement in objective value mainly translates into shorter delays and increased waiting time. The handling and fuel costs remain similar. The vessel time windows or port calls are already

Table 4.12: Average operational costs and cost variation across instances with different quay segment lengths. All instances are run for one hour. The costs are in thousands of USD.

Segment length (m)	Waiting	Handling	Delay	Fuel	Total	Penalty	Iterations x1000
10	390	537	3210	1274	5411	0.04	8
20	368	537	3246	1275	5426	0.05	15
40	242	541	3449	1276	5509	0.09	18
80	241	542	3910	1275	5968	0.14	35

pre-planned, considering a low sailing speed. This, together with the fact that fuel costs account for a large part of the total costs, results in that ships already sailing at the slowest speed in most of the solutions (see Table 4.14).

Table 4.13: Operational costs for instances grouped by different amounts of external ships per port.

External ships per port	Waiting	Handling	Delay	Fuel	Total	Penalty
0	451	538	2365	1278	4632	0.06
5	287	538	3022	1275	5122	0.04
10	301	538	3227	1275	5341	0.05
20	461	538	2718	1278	8300	0.11

Another operational aspect we inspect is the impact of the external ships in the planning process. We solve the problem for instances with a different number of external ships per port, from none to twenty ships per port. The results are summarized in Table 4.13. The results support the rationale that an increased number of external ships per port results in a more congested berth allocation and, as a result, higher operational costs. In this case, the port congestion is reflected in the *Penalty* column, which indicates the average number of port visits per instance exceeding the latest finish time. Since this type of delay is heavily penalized, improvements in this aspect significantly impact the objective value. These results also indicate that the level of impact of this type of collaborative problem can increase significantly when more ships are involved. When more ships collaborate, their potential joint savings increase and the terminal has more planning flexibility.

As mentioned previously, fuel consumption is the main cost driver for carriers. Fuel prices have fluctuated significantly in the last two years due to the global socio-economical and political situation. We consider that ships use a very low sulfur fuel oil (VLSFO) with an estimated price of 500 USD per metric ton.

However, the prices of this fuel have ranged between 200 and 1100 USD per metric ton in the last two years. Therefore, we have also tested our method using a fuel price of 200 and 1100 USD per metric ton (Ship & Bunker, 2022)

Table 4.14: Average fuel consumption per ship and sailing speed based on different fuel prices

Number of ships	**Fuel price (USD/metric tonne)**					
	200		**500**		**1100**	
	Fuel	*Speed*	*Fuel*	*Speed*	*Fuel*	*Speed*
30	51.04	17.12	50.53	17.04	50.37	17.01
50	51.50	17.10	51.11	17.04	50.99	17.02
70	52.14	17.19	51.35	17.08	51.07	17.03
Average	51.56	17.14	51.00	17.05	50.81	17.02

Table 4.14 shows the average fuel consumption (*Fuel*) in metric tonne per ship and sailing speed (*Speed*) in knots, grouped by instances with the same number of ships. We observe that the average consumption can increase by more than half a metric tonne when the fuel price decreases from 500 USD per tonne to 200 USD per tonne. This difference is more prominent in instances with a large number of ships, where more ships sail marginally faster to arrive earlier at the next port to get a better service. However, when the fuel price increases above 500 USD per metric tonne, the reductions in fuel consumption are relatively small. The main explanation for this is due to the low sailing speeds in general. The fuel costs already account for a large part of the operational costs, and the solutions indicate that ships sail close to the slowest speed of 17 knots in most cases. We observe a slight increase in average sailing speed when the fuel price is low, but the size of the instance does not have an impact on the speed. A similar sensitivity analysis performed by Venturini et al. (2017) indicated a similar behavior.

4.6 Conclusions

In this work, we address an emerging problem in maritime collaborative logistics that integrates the operations of both shipping carriers and terminal operators. We present both a new MIP formulation for the multi-port continuous berth allocation problem with speed optimization, and an ALNS algorithm to solve it. The ALNS algorithm takes advantage of a diverse set of tailored insertion and removal methods. It guides the algorithm by prioritizing the better-performing methods. The modular characteristic of the algorithm could be exploited to develop a decision support tool for terminal operators, where the operators' experience can lead to new tailored operators. Furthermore, in terms of com-

putational performance, the heuristic method is able to find high-quality solutions to larger instances than the ones studied in the literature and outperforms commercial solvers such as CPLEX. We also study the practical impact of the problem in terms of operational costs for the carriers and terminal operators and analyze the resulting quality of the berth plans and sailing speeds. We conclude that engaging in this type of collaboration can result in overall cost reductions for the stakeholders and also benefits to the environment due to the potential lower fuel consumption.

Some aspects of this study remain as future work or research direction. Regarding the solution method, the insertion operators are the main bottleneck in terms of computational complexity. One could explore simpler insertion operators or other heuristic variants. Studying the scalability of the method in more detail could be relevant. There is no doubt that the heuristic method scales better than CPLEX, and results in small instances indicate that the ALNS achieves near-optimal solutions. For larger instances, the optimality gap of CPLEX increases significantly, and the lower bound becomes impractical. Finally, incorporating practical aspects such as transhipments or disruptions management is an attractive research direction. We envision the use of frameworks such as stochastic programming to tackle this type of problem. All in all, this type of study highlights the potential impact of collaborative logistics and the value of integration in the transportation sector.

Acknowledgements: The authors thank the Danish Maritime Fund for supporting this work.

References

Ansótegui, C., Pon, J., and Sellmann, M. (2021). Boosting evolutionary algorithm configuration. *Annals of Mathematics and Artificial Intelligence*.

APM Terminals (2022a). APM Terminals Bremerhaven NTB. https://www.apmterminals.com/en/bremerhaven/practical-information/practical-information. Accessed: 2022-11-4.

APM Terminals (2022b). APM Terminals Maasvlakte II. https://www.apmterminals.com/en/maasvlakte/about/our-terminal. Accessed: 2022-09-15.

Bierwirth, C. and Meisel, F. (2015). A follow-up survey of berth allocation and quay crane scheduling problems in container terminals. *European Journal of Operational Research*, 244(3):12689, 675–689.

Bräysy, O. (2003). A reactive variable neighborhood search for the vehicle-routing problem with time windows. *Informs Journal on Computing*, 15(4):347–368.

Carlo, H. J., Vis, I. F., and Roodbergen, K. J. (2014). Transport operations in container terminals: Literature overview, trends, research directions and classification scheme. *European Journal of Operational Research*, 236(1):1–13.

Cheimanoff, N., Fontane, F., Kitri, M. N., and Tchernev, N. (2022). Exact and heuristic methods for the integrated berth allocation and specific time-invariant quay crane assignment problems. *Computers and Operations Research*, 141:105695.

Cordeau, J. F., Laporte, G., Legato, P., and Moccia, L. (2005). Models and tabu search heuristics for the berth-allocation problem. *Transportation Science*, 39(4):526–538.

Du, Y., Chen, Q., Quan, X., Long, L., and Fung, R. Y. (2011). Berth allocation considering fuel consumption and vessel emissions. *Transportation Research Part E: Logistics and Transportation Review*, 47(6):1021–1037.

Dulebenets, M. A. (2018). A comprehensive multi-objective optimization model for the vessel scheduling problem in liner shipping. *International Journal of Production Economics*, 196:293–318.

Dulebenets, M. A. (2019). Minimizing the total liner shipping route service costs via application of an efficient collaborative agreement. *Ieee Transactions on Intelligent Transportation Systems*, 20(1):8315131.

Dulebenets, M. A., Golias, M. M., and Mishra, S. (2018). A collaborative agreement for berth allocation under excessive demand. *Engineering Applications of Artificial Intelligence*, 69:76–92.

Dulebenets, M. A., Pasha, J., Abioye, O. F., and Kavoosi, M. (2019). Vessel scheduling in liner shipping: a critical literature review and future research needs. *Flexible Services and Manufacturing Journal*, 33(1):43–106.

Eurogate (2022). Eurogate Hamburg. http://www1.eurogate.de/en/EUROGATE/Terminals/Hamburg. Accessed: 2022-11-4.

Fagerholt, K. (2001). Ship scheduling with soft time windows: An optimisation based approach. *European Journal of Operational Research*, 131(3):559–571.

Fagerholt, K., Laporte, G., and Norstad, I. (2010). Reducing fuel emissions by optimizing speed on shipping routes. *Journal of the Operational Research Society*, 61(3):523–529.

Glover, F. (1992). New ejection chain and alternating path methods for traveling salesman problems. *Computer Science and Operations Research. New Developments in Their Interfaces*, pages 491–508.

Guan, Y. and Cheung, R. K. (2005). The berth allocation problem: Models and solution methods. *Container Terminals and Automated Transport Systems: Logistics Control Issues and Quantitative Decision Support*, pages 141–158.

Guo, L., Zheng, J., Du, H., Du, J., and Zhu, Z. (2022). The berth assignment and allocation problem considering cooperative liner carriers. *Transportation Research Part E: Logistics and Transportation Review*, 164:102793.

Hellsten, E. O., Sacramento Lechado, D., and Pisinger, D. (2020). An adaptive large neighbourhood search heuristic for routing and scheduling feeder vessels in multi-terminal ports. *European Journal of Operational Research*, 287(2):682–698.

Imai, A., Sun, X., Nishimura, E., and Papadimitriou, S. (2005). Berth allocation in a container port: using a continuous location space approach. *Transportation Research Part B-methodological*, 39(3):199–221.

IMO (2018). Initial IMO strategy on reduction of GHG emissions from ships. Technical Report MEPC.304(72), International Maritime Organization. (Accessed on 04.05.2020).

Iris, C. and Lam, J. S. L. (2018). Models for continuous berth allocation and quay crane assignment: Computational comparison. *Ieee International Conference on Industrial Engineering and Engineering Management*, 2017-:374–378.

Iris, C., Pacino, D., and Røpke, S. (2017). Improved formulations and an adaptive large neighborhood search heuristic for the integrated berth allocation and quay crane assignment problem. *Transportation Research. Part E: Logistics and Transportation Review*, 105:123–147.

Kim, K. H. and Moon, K. C. (2003). Berth scheduling by simulated annealing. *Transportation Research Part B: Methodological*, 37(6):541–560.

Kordić, S., Davidović, T., Kovač, N., and Dragović, B. (2016). Combinatorial approach to exactly solving discrete and hybrid berth allocation problem. *Applied Mathematical Modelling*, 40(21-22):8952–8973.

Lalla-Ruiz, E., Melián-Batista, B., and Moreno-Vega, J. M. (2016). A cooperative search for berth scheduling. *Knowledge Engineering Review*, 31(05):498–507.

Lim, A. (1998). The berth planning problem. *Operations Research Letters*, 22(2-3):105–110.

Lyu, X., Negenborn, R. R., Shi, X., and Schulte, F. (2022). A collaborative berth planning approach for disruption recovery. *Ieee Open Journal of Intelligent Transportation Systems*, 3:153–164.

Marine Traffic (2023). Port Calls Data. https://www.marinetraffic.com/en/data/. Accessed: 2023-01-14.

Martin-Iradi, B., Pacino, D., and Ropke, S. (2022a). The multi-port continuous berth allocation problem with speed optimization. In de Armas, J., Ramalhinho, H., and Voß, S., editors, *Computational Logistics*, pages 31–43, Cham. Springer International Publishing.

Martin-Iradi, B., Pacino, D., and Ropke, S. (2022b). The multiport berth allocation problem with speed optimization: Exact methods and a cooperative game analysis. *Transportation Science*, 56(4):972–999.

Meisel, F. and Bierwirth, C. (2009). Heuristics for the integration of crane productivity in the berth allocation problem. *Transportation Research Part E: Logistics and Transportation Review*, 45(1):196–209.

Potvin, J. and Rousseau, J. (1993). A parallel route building algorithm for the vehicle-routing and scheduling problem with time windows. *European Journal of Operational Research*, 66(3):331–340.

Rego, C. (1998). A subpath ejection method for the vehicle routing problem. *Management Science*, 44(10):1447–1459.

Ropke, S. and Pisinger, D. (2006). An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation Science*, 40(4):455–472.

Shaw, P. (1998). Using constraint programming and local search methods to solve vehicle routing problems. *Principles and Practice of Constraint Programming - Cp98. 4th International Conference, Cp98. Proceedings*, pages 417–31.

Ship & Bunker (2022). Global Average Bunker Price - VLSFO. https://shipandbunker.com/prices/av/global/av-g20-global-20-ports-average. Accessed: 2022-12-18.

Steenken, D., Voß, S., and Stahlbock, R. (2004). Container terminal operation and operations research - a classification and literature review. *Or Spectrum*, 26(1):3–49.

Sun, B., Niu, B., Xu, H., and Ying, W. (2018). Cooperative optimization for port and shipping line with unpredictable disturbance consideration. *Icnc-fskd 2018 - 14th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery*, pages 8686901, 113–118.

UNCTAD (2020). Review of maritime transport 2020. Technical report, UNCTAD.

Venturini, G., Iris, C., Kontovas, C. A., and Larsen, A. (2017). The multi-port berth allocation problem with speed optimization and emission considerations. *Transportation Research. Part D: Transport and Environment*, 54:142–159.

Wang, S., Liu, Z., and Qu, X. (2015). Collaborative mechanisms for berth allocation. *Advanced Engineering Informatics*, 29(3):572, 332–338.

Yu, J., Tang, G., and Song, X. (2022). Collaboration of vessel speed optimization with berth allocation and quay crane assignment considering vessel service differentiation. *Transportation Research Part E: Logistics and Transportation Review*, 160:102651.

Part III

Microtransit

Chapter 5

Design and operation of on-demand microtransit systems

Bernardo Martin-Iradi[a], Kayla Cummings[b], and Alexandre Jacquillat[c]

[a]DTU Management, Technical University of Denmark, 2800 Kongens Lyngby, Denmark

[b]Operations Research Center, Massachusetts Institute of Technology, 02142 Cambridge, MA, United States

[c]Sloan School of Management, Massachusetts Institute of Technology, 02142 Cambridge, MA, United States

Status: To be submitted to an international peer-review journal.

Abstract: Demand-responsive microtransit offers opportunities to enhance urban mobility by combining the reliability of public transit and the flexibility of ride-sharing. This paper optimizes the design and operations of a microtransit system that relies on *reference lines* and that performs on-demand deviations in response to passenger requests. We formulate a *Microtransit Network Design model for Vehicle Routing (MiND-VRP)* via two-stage stochastic optimization with a tight second-stage structure. This formulation leverages a novel subpath-based representation of microtransit in a load-expanded network, which optimizes on-demand deviations between reference stops while keeping track of vehicle occupancy in between. We develop a solution algorithm combining Benders decomposition, column generation and a tailored label setting algorithm. Using real-world data from Manhattan, our method scales to very large instances arising in practice, with dozens of lines and hundreds of reference stops. Comparisons with transit and ride-sharing offerings suggest that demand-responsive microtransit can provide win-win outcomes toward efficient and sustainable mobility: higher demand coverage, better level of service, and smaller environmental footprint.

Keywords: Public transit, microtransit, stochastic optimization

5.1 Introduction

Major cities are facing critical challenges to meet mobility needs and environmental impact targets. As population grows and urbanization accelerates, so

do travel demand and e-commerce deliveries. INRIX (2022) estimated the annual impact of traffic congestion in the United States is around a hundred hours per driver and hundreds of billions of dollars in total. Urban transportation also remains a major contributor to greenhouse gas emissions (US Environmental Protection Agency, 2018) and to socio-economic inequalities (Wellman, 2014). Transit systems play a pivotal role in addressing these challenges. Yet, their static infrastructure offers limited service flexibility, leading to a decline in transit ridership and the emergence of transit deserts (The Economist, 2018; Allen, 2017). The COVID-19 pandemic reinforced the need to adapt to changing mobility needs. Thus, new technologies and policies are needed to promote effective, sustainable, and equitable mobility.

Shared mobility systems introduce the most important disruption to urban mobility. McKinsey & Co. (2021) estimates that the ride-sharing market tripled between 2016 and 2019, with 40 million combined trips booked every day on Uber and Lyft. These staggering numbers suggest that flexible, on-demand services meet a critical need of urban travelers and partially displace transit demand as a result (Gehrke and Reardon, 2018). However, ride-sharing remains insufficient to fundamentally alter the commuting landscape by itself due to high fares and their contributions to urban congestion (Mangrum and Molnar, 2017; Agarwal et al., 2023).

This context creates opportunities to design hybrid *microtransit* services, defined by the US Department of Transportation (2016) as "privately owned and operated shared transportation system(s) that can offer fixed routes and schedules, as well as flexible routes and on-demand scheduling." In general terms, microtransit brings the digital capabilities of ride-sharing into the realm of public transit to embed flexibility into existing transit offerings or to serve *transit deserts,* i.e. areas where transit is unavailable. Microtransit has already been piloted in several cities (OECD, 2017; Eno Center for Transportation, 2018; Hernandez, 2018; Boston Consulting Group, 2019). These systems experiment with new ways to combine the reliability of public transit and the flexibility of ride-sharing, by merging advance planning with on-demand operations for high-capacity vehicle fleets. As a result, microtransit has the potential to increase the responsiveness of urban mobility at affordable price points. Yet, to be successful, microtransit systems require dedicated analytics and thoughtful design to achieve low costs and high levels of service (McKinsey & Co., 2018).

This paper develops models and algorithms to support the ongoing transition toward demand-responsive microtransit and to evaluate its potential for enhancing urban mobility systems. To balance predictability and flexibility, we conceptualize microtransit as a system that operates in two steps. At the strategic level, the microtransit provider advertises *reference lines*, each defined as an ordered set of stops that a vehicle will visit at designated times. At the operational

level, the microtransit provider may deviate from the reference lines to better serve on-demand passenger requests. This paper addresses the problems of network design (which reference lines to operate), service scheduling (with which frequency and timetable to operate on each reference line), and vehicle routing (how to operate vehicles in response to on-demand passenger requests).

From a technical standpoint, the design of microtransit systems combines difficulties of public transit and on-demand transportation. Public transit involves challenging planning problems to provide high service coverage under budget restrictions. At the other extreme, on-demand transportation involves challenging operating problems to devise vehicle routes in response to real-time rider demand. In-between, microtransit systems combines strategic planning questions and real-time operating questions, thus requiring customized modeling and algorithmic frameworks.

Our first contribution is to formulate a *Microtransit Network Design model for the Vehicle Routing Problem (MiND-VRP)* to support the design and on-demand operations of microtransit systems. The model exhibits a two-stage stochastic optimization structure under demand uncertainty. The first stage selects reference lines and corresponding frequencies. The second stage captures on-demand routing deviations to serve passenger requests over a discrete set of scenarios characterizing passenger demand. The model features a bi-objective structure to maximize both ridership and level of service, comprising passengers' wait times, walking distances, and arrival delays. This problem combines two challenging discrete optimization problems: a first-stage network design structure and a second-stage capacitated routing structure with time windows.

To retain a tight second-stage formulation, we propose a subpath-based formulation of microtransit operations in a novel load-expanded network. Each node identifies the reference stop and the vehicle load, and each subpath-based arc represents an on-demand deviation between reference stops. We show that our subpath-based variables enable a more effective decomposition than direct segment-based variables (between pickup locations) and path-based variables (connecting the origin to the destination of each transit line). As compared to the direct segment-based formulation, our subpath-based formulation integrates time window requirements in the definition of the subpaths so the problem can be formulated in a load-expanded network rather than a time-load-expanded network. As compared to the path-based formulation, our subpath-based formulation drastically quells the rate of exponential growth in the number of variables.

The second contribution of this paper is to design a decomposition algorithm that combines column generation and Benders decomposition to solve large-scale instances of the MiND-VRP arising in practice. The Benders decomposition

module exploits the block-diagonal structure of the two-stage stochastic optimization formulation, iterating between the first-stage network design model and the second-stage routing problems. The column generation module solves each Benders subproblem by iteratively adding subpaths between reference stops. We develop a label-setting algorithm to generate subpaths of minimal reduced cost while keeping track of vehicle load and level of service. We also propose exact and heuristic acceleration strategies. Ultimately, our approach is amenable to effective decomposition by leveraging a subpath-based structure in load-expanded networks and a corresponding subpath-generation solution procedure.

Our third contribution is to demonstrate the scalability of our modeling and algorithmic approach via extensive computational experiments. We develop a real-world experimental setup in Manhattan using data from the Metropolitan Transportation Authority (2022) to define candidate transit lines and data from the NYC Taxi & Limousine Commission (2021) to generate demand scenarios. Our subpath-based formulation scales much better than the direct segment-based baseline and than the path-based baseline: it terminates much faster in medium-scale instances, and it can solve large-scale instances when the two benchmarks fail to even return feasible solutions. We also show the combined benefits of our Benders decomposition structure, our column generation scheme and our label-setting algorithm toward solving large-scale instances of the problem. As a result, our algorithm yields high-quality solutions in large-scale instances in the full Manhattan network with dozens of lines and hundreds of reference stops. These results provide high-quality solutions in a setting of comparable size to recent single-stage network design models in the fixed-route transit literature, while capturing demand uncertainty and second-stage microtransit operations; moreover, our paper considers a much larger setting than single-stage vehicle routing formulations in the microtransit literature, while capturing demand uncertainty and first-stage network design. In summary, our method provides results at highly competitive scale in both branches of literature, but also integrates the tactical on-demand operations into strategic network design.

The fourth contribution is to show that microtransit can provide win-win outcomes toward efficient and sustainable urban mobility. We perform a comprehensive assessment of the microtransit system against existing urban mobility solutions. As compared to ride-sharing, microtransit can consolidate demand into high-capacity vehicles under high demand or low capacity, resulting in fewer vehicle miles traveled than single-occupancy ride-sharing and a comparable level of service to high-capacity ride-sharing. As compared to fixed-route transit, microtransit can serve 25-30% more passengers with the same budget of lines, by leveraging operating flexibility to deviate from the reference line up to 30% of the time. Ultimately, the microtransit system under consideration can achieve Pareto improvements as compared to fixed-route transit: higher demand cover-

age, lower operating costs, and smaller distance traveled along with concomitant environmental benefits.

5.2 Literature review

The demand-responsive microtransit system considered in this paper is related to the literature on transit planning, ride-sharing, and on-demand transit.

Transit network design

Transit has received extensive attention from the operations research community (see, e.g., Magnanti and Wong, 1984; Desaulniers and Hickman, 2007). Canonical problems include frequency planning, timetabling, and disruption recovery (see, e.g., Nourbakhsh and Ouyang, 2012; Barrena et al., 2014; Ortega et al., 2018; Sun et al., 2022, for recent contributions). Our paper relates to the upstream problem of designing transit lines to maximize demand coverage, maximize connectivity, and adhere to budget restrictions. Due to its complexity, the problem has often been solved with heuristics (see, e.g., Ceder and Wilson, 1986; Baaj and Mahmassani, 1995; Cipriani et al., 2012; Walteros et al., 2015). Among exact methods, Wan and Lo (2003) used mixed-integer optimization, Barra et al. (2007) used constraint programming, and Marín and Jaramillo (2009) used Benders decomposition. However, these formulations could only scale to small instances comprising 10-25 stops. Borndörfer et al. (2007) and Borndörfer and Karbstein (2012) proposed a column generation algorithm to solve an incremental network design problem. Bertsimas et al. (2021) addressed a comprehensive network design problem, also using column generation; they generated solutions in large-scale networks with hundreds of stops and thousands of edges.

Demand-responsive microtransit combines reference transit lines and on-demand routing deviations. Our first-stage model therefore relates to transit network design, but our formulation differs via the second-stage model. Technically, our framework exhibits a column-dependent row structure due to linking relationships between first-stage design decisions and second-stage operating decisions, which severely complicates the use of column generation (as in Borndörfer et al., 2007; Bertsimas et al., 2021). Instead, we design a pre-processing procedure to candidate lines, and select reference lines among those in our first-stage problem—we still employ column generation in the second stage to generate subpaths characterizing on-demand deviations. This pre-processing approach has been widely used in the literature (see, e.g., Ceder and Wilson, 1986); it is also common in practice to define a set of candidate lines based on domain knowledge and practical requirements.

Ride-sharing

An extensive line of work has characterized optimal or asymptotically optimal policies in ride-sharing to support vehicle-customer matching, pricing, vehicle rebalancing, etc. (see, e.g., Braverman et al., 2019; Özkan and Ward, 2020; Balseiro et al., 2021). Our second-stage problem optimizes vehicle routes to serve discrete passenger requests, which relates to the vehicle routing problem with time windows and capacitated vehicles—a highly challenging integer optimization problem (Baldacci et al., 2011). We model it with a network representation of routing operations to retain a tight second-stage formulation. This relates to the ride-shareability network from Santi et al. (2014), to the vehicle-shareability network from Vazifeh et al. (2018) and to the request-trip-vehicle graph network from Alonso-Mora et al. (2017). Similarly, Bertsimas et al. (2019) optimized ride-sharing operations via a network flow formulation on a vehicle-sharing network. Zhang et al. (2022) optimized ride-pooling operations with vehicle-customer coordination in a network where arcs refer to vehicle trips between two points where the vehicle is empty. Our paper contributes a tailored load-expanded subpath-based network representation of microtransit operations by leveraging the reliance on reference stops, which we use to optimize on-demand deviations.

On-demand transit

Ride-sharing has motivated digital on-demand transit solutions. Salazar et al. (2018); Ma et al. (2019) and Stiglic et al. (2018) optimized inter-modal operations for ride-sharing and public transit to provide complementary mobility options, for instance by using ride-sharing for first- and last-mile transportation and public transit for increasing accessibility. Cummings et al. (2023) designed a pricing alliance between transit agencies and ride-sharing providers to enhance service coverage and decrease the reliance on single-occupancy vehicles. Shen et al. (2018) used agent-based simulation and Wei et al. (2022) used mixed-integer non-linear optimization to re-purpose transit resources from areas that are served well by ride-sharing toward areas where transit is more competitive. Steiner and Irnich (2020) developed an exact branch-and-price algorithm and Banerjee et al. (2021) proposed an approximate randomized rounding scheme to design integrated systems combining fixed-line transit and on-demand mobility.

In contrast, our paper designs a hybrid demand-responsive microtransit system that features some static components akin to public transit and some dynamic components akin to ride-sharing. One concept operates high-capacity transit vehicles on demand (as in Alonso-Mora et al., 2017). Daganzo and Ouyang (2019) proposed a queuing-theoretic framework to compare transit, ride-sharing, dial-a-ride and ride-pooling. Silva et al. (2022) developed a Markovian model of on-demand transit operations, using continuous routing approximations. Ex-

tensions include horizontal collaboration between demand-responsive transit providers (Angelelli et al., 2022) and hybrid systems combining fixed-line and demand-responsive services (Azadeh et al., 2022), both solved with heuristics.

More closely related to our paper, mobility allowance shuttle transit (MAST) systems rely on a fixed route, but vehicles can perform on-demand deviations within a predetermined region. Quadrifoglio et al. (2007, 2008) optimized these deviations for a single vehicle via a pickup-and-delivery problem, solved with an insertion heuristic and column generation. Quadrifoglio et al. (2006) and Zhao and Dessouky (2008) quantified the trade-offs between frequency, deviation magnitude, and service levels. Galarza Montenegro et al. (2021, 2022) proposed a related concept in which transit vehicles can visit or skip stops based on on-demand requests, and also optimized operations using heuristics and column generation. These systems offer some predictability through a fixed route, a predetermined schedule, and mandatory stops, as well as some flexibility through on-demand adjustments to vehicle routing, service timetabling and service frequency. Yet, demand-responsive operations remain complex: existing methods scale up to 1-5 vehicles and 10-50 stops.

Our paper contributes to this literature in four ways. For the first time, we tackle the strategic problem of network design in demand-responsive microtransit, by jointly optimizing reference lines and on-demand deviations. Also for the first time, we capture uncertainty and variability in microtransit demand via a two-stage stochastic optimization formulation. We also develop a solution algorithm combining Benders decomposition, column generation, and a tailored label-setting algorithm. Our modeling and algorithmic approach scales to the full system of Manhattan with over 100 vehicles, hundreds of reference stops, over 1,000 actual stops for on-demand deviations, thousands of passenger requests, and 50 scenarios. Finally, we report practical results to evaluate the potential of demand-responsive microtransit toward efficient, flexible, and sustainable transit.

5.3 Microtransit network design for vehicle routing

The MiND-VRP model optimizes the design and operations of demand-responsive microtransit systems. We assume a homogeneous fleet of vehicles, each with capacity C. We consider a setting in which all passengers share the same destination. We can therefore view the problem as a microtransit vehicle routing problem, as opposed to microtransit-oriented dial-a-ride in which passengers would request transportation from an origin to a destination. The design phase defines *reference lines*, each comprising a list of *reference stops*. As in public transit,

the reference lines enable passengers to plan their travels and service providers to consolidate passenger demand into high-capacity vehicles. The operations phase involves on-demand deviations within allowable margins to increase demand coverage and to provide convenient mobility options. This microtransit system therefore combines network design and frequency planning elements of public transit as well as vehicle routing elements of on-demand transportation.

We formulate the MiND-VRP as a two-stage stochastic optimization model. The first stage designs reference lines and sets corresponding frequencies (Section 5.3.1) The second stage leverages subpath-based optimization in a load-expanded network to capture on-demand deviations in response to passenger demand (Section 5.3.2). We provide the full MiND-VRP formulation in Section 5.3.3, and we compare it to segment-based and path-based benchmarks in Section 5.3.4.

5.3.1 First-stage problem: network design and frequency planning

The first-stage problem defines a microtransit schedule, consisting of reference lines and corresponding service frequencies. We refer to a reference line and departure time pair as a *reference trip.* Each reference trip defines the scheduled arrival time at each reference stop. Vehicles will be required to visit some reference stops at the scheduled times, but they will also be allowed to visit other pickup locations to better serve on-demand passenger requests (Section 5.3.2). Operations unfold in a roadway network. We denote by $\mathcal{N}$ the set of possible stopping location for the vehicle, including all candidate reference stops and all possible pickup locations for passengers.

We pre-process candidate reference lines in a set $\mathcal{L}$. Let h_ℓ denote the cost to operate one trip of line $\ell \in \mathcal{L}$. Let $\mathcal{T}_\ell$ denote the times when a vehicle can depart from the first stop in line $\ell \in \mathcal{L}$. Each tuple (ℓ, t) consequently defines a reference trip. We define the following variables:

$$x_{\ell t} = \begin{cases} 1 & \text{reference trip } (\ell, t) \text{ is selected, for } \ell \in \mathcal{L} \text{ and } t \in \mathcal{T}_\ell, \\ 0 & \text{otherwise.} \end{cases} \tag{5.1}$$

Let $\mathcal{I}_\ell$ index reference stops in line ℓ, of cardinality $I_\ell = |\mathcal{I}_\ell|$; let $\mathcal{I}_\ell^{(i)}$ refer to the i^{th} reference stop in the line. For a given reference trip $(\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell$, each reference stop $i \in \mathcal{I}_\ell$ is scheduled to be visited by a vehicle at time $T_{\ell t}(i)$. All reference lines share the same final stop $\mathcal{I}_\ell^{(I_\ell)}$.

First-stage decisions also assign passengers to reference trips. Let $\mathcal{P}$ define *passenger requests*, each with an origin $o(p) \in \mathcal{N}$ and a request time t_p^{req}. Each

passenger request $p \in \mathcal{P}$ can be served by a subset of reference trips $\mathcal{M}_p \subseteq \mathcal{L} \times \mathcal{T}_\ell$. We define the following variables:

$$z_{p\ell t} = \begin{cases} 1 & \text{if passenger type } p \in \mathcal{P} \text{ is assigned to trip } (\ell, t) \in \mathcal{M}_p, \\ 0 & \text{otherwise.} \end{cases} \tag{5.2}$$

Due to demand uncertainty, first-stage passenger assignment decisions are modeled separately from second-stage passenger pickup decisions. The $\boldsymbol{z}$ decisions simply provide passengers with a frame of reference before they request a ride. Technically, they also guide the first-stage problem toward covering passenger demand without exclusively relying on second-stage recourse.

5.3.2 Second-stage problem: on-demand deviations

In the second stage, the microtransit operator optimizes routing deviations from reference trips in response to on-demand passenger requests. To capture demand uncertainty, we define a set of demand scenarios $\mathcal{S}$. Let D_{ps} denote the number of passengers in request $p \in \mathcal{P}$ in scenario $s \in \mathcal{S}$.

We restrict on-demand deviations to a spatial-temporal neighborhood of the reference line for the microtransit system to retain predictability based on the reference schedule. Specifically, vehicles may skip up to K reference stops in a row. The parameter K captures the trade-off between adherence to the reference schedule versus on-demand deviations. As in Galarza Montenegro et al. (2021, 2022), vehicles must stay within a maximum distance from the reference line between reference stops, and they must respect the scheduled arrival times at the visited reference stops. The reference schedules include slack between reference stops to allow for such deviations.

The second-stage problem involves capacitated vehicle routing with time windows for each reference trip in each scenario. We formulate it using a subpath-based load-expanded network, where *load* refers to the number of passengers in the vehicle.

Subpaths

Subpath variables characterize deviations between reference stops. These subpaths are longer than road segments between pickup locations, but they are shorter than paths that encapsulate full trips from the start of the line to the end (see Section 5.3.4). This model follows recent dial-a-ride formulations from Alyasiry et al. (2019); Zhang et al. (2022), which define subpaths from a point where the vehicle is empty to another. Our subpath variables introduce a new decomposition tailored to the microtransit system by leveraging the reliance on the reference stops.

Let $\mathcal{R}_{\ell st}$ denote the set of subpaths for a given trip $(\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell$ and demand scenario $s \in \mathcal{S}$. A subpath $r \in \mathcal{R}_{\ell st}$ is uniquely identified by its starting point $u_r \in \mathcal{I}_\ell$, ending point $v_r \in \mathcal{I}_\ell$, and the passenger requests $\mathcal{P}_r \subseteq \mathcal{P}$ that it serves. Any subpath $r \in \mathcal{R}_{\ell st}$ must satisfy load limits $\sum_{p \in \mathcal{P}_r} D_{ps} \leq C$; the travel time on subpath r must not exceed $T_{\ell t}(v_r) - T_{\ell t}(u_r)$; and each subpath can skip up to K reference stops. For each reference trip and each scenario, the second-stage problem selects a sequence of subpaths that define a valid trip that (i) starts at the origin of the reference line, ends at its destination, and maintains flow balance in between; (ii) does not pick up more than C passengers overall; and (iii) only picks up passengers that have been assigned to the trip.

Load-expanded subpath network

We represent routing operations in a load-expanded network. Each node tracks the reference stop and the vehicle load, and each arc encapsulates a subpath between reference stops along with the number of onboarding passengers. In this network, flow balance constraints capture both physical flow balance and vehicle capacity constraints (requirements (i) and (ii) above). It would also be possible to formulate the problem in a flat network, with nodes solely tracking reference stops and arcs solely capturing physical subpaths. In this alternative representation, flow balance constraints would merely capture physical flow balance, so we would have to enforce vehicle capacity via big-M constraints. In other words, our load-expanded network formulation involves more variables, but the second-stage formulation is tighter as a result. Figure 5.1 illustrates the construction of our load-expanded subpath network.

Mathematically, let $\mathcal{C} = \{0, 1, \cdots, C\}$ store all valid vehicle loads. For each reference trip $(\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell$ and scenario $s \in \mathcal{S}$, we denote the load-expanded network by $(\mathcal{V}_{\ell st}, \mathcal{A}_{\ell st})$. Each node $n \in \mathcal{V}_{\ell st}$ corresponds to a tuple (k_n, c_n), where $k_n \in \mathcal{I}_\ell$ tracks the reference stop and $c_n \in \mathcal{C}$ tracks the load. We include a dummy sink node $v_{\ell st}$ representing the end of a trip. The source node of the trip is $u_{\ell st} = (\mathcal{I}_\ell^{(1)}, 0)$. Each arc $a \in \mathcal{A}_{\ell st}$ connects node $start(a) \in \mathcal{V}_{\ell st}$ and node $end(a) \in \mathcal{V}_{\ell st}$ in the load-expanded network. We separate the arc set $\mathcal{A}_{\ell st} = \bigcup_{r \in \mathcal{R}_{\ell st}} \mathcal{A}_r \cup \mathcal{A}^v_{\ell st}$ into two types of arcs: traveling arcs and idling arcs. Each arc $a \in \mathcal{A}_r$ defines the corresponding subpath $r \in \mathcal{R}_{\ell st}$ by tracking its starting point u_r, its ending point v_r, and its incremental load $\sum_{p \in \mathcal{P}_r} D_{ps}$:

$$\mathcal{A}_r = \left\{ (n, m) \in \mathcal{V}_{\ell st} \times \mathcal{V}_{\ell st} \,:\, k_n = u_r, k_m = v_r, c_m - c_n = \sum_{p \in \mathcal{P}_r} D_{ps} \right\}, \quad \forall r \in \mathcal{R}_{\ell st}. \tag{5.3}$$

For a given arc $a \in \bigcup_{r \in \mathcal{R}_{\ell st}} \mathcal{A}_r$, we denote its corresponding physical subpath by $r(a) \in \mathcal{R}_{\ell st}$.

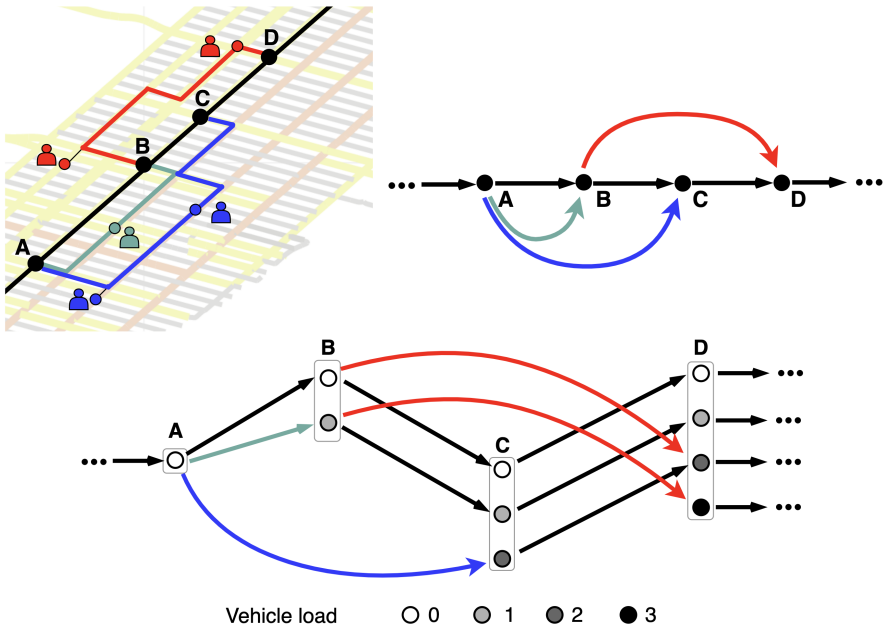


Figure 5.1: Representation of deviations on the load-expanded subpath network. *Top left:* Physical network with reference stops A, B, C, and D, and 3 reference line deviations. *Top right:* Subpath representation of the deviations. *Bottom:* Load-expanded network representation.

Next, each terminating arc $a \in \mathcal{A}^v_{\ell st}$ connects the line's destination to the dummy sink node:

$$\mathcal{A}^v_{\ell st} = \{(n, m) \in \mathcal{V}_{\ell st} \times \mathcal{V}_{\ell st} \,:\, k_n = \mathcal{I}_\ell^{(I_\ell)}, m = v_{\ell st}\}. \tag{5.4}$$

Once constructed, the load-expanded network can be pruned to exclude intermediate nodes in $\mathcal{V}_{\ell st}$ with no incoming or outgoing arcs, and to exclude all corresponding arcs in $\mathcal{A}_{\ell st}$.

Finally, our second-stage decisions select subpaths in the load-expanded networks, which each define on-demand deviations and passenger pickups for each vehicle trip in each demand scenario:

$$y_a = \begin{cases} 1 & \text{if arc } a \text{ is selected, for } (\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell,\ s \in \mathcal{S}, a \in \mathcal{A}_{\ell st}, \\ 0 & \text{otherwise.} \end{cases} \tag{5.5}$$

Passenger service

The microtransit system performs on-demand deviations to improve passenger experience, which includes picking up passengers closer to their requested location or their requested times. This is formalized in the MiND-VRP by restricting the amount of waiting and walking that any passenger is willing to accept, and then by maximizing level of service.

Accordingly, passengers can only be picked up if they have enough time to walk from their origins to the pickup location and if the wait time does not exceed a maximum allowable limit. In other words, each load-expanded subpath arc must satisfy two conditions for its passenger pickup set:

- The distance from the passengers' origins to the pickup locations must lie below a maximum allowable distance Ω. Let $\omega_{o,d}$ denote the walking distance from o to d. Pickup location $i \in \mathcal{N}$ is only acceptable to passenger $p \in \mathcal{P}$ if $\omega_{o(p),i} \leq \Omega$.
- Passengers' wait times at the pickup locations must lie below a maximum allowable wait time Ψ. Let $\psi_{o,d}$ be the walking time from o to d. Passengers can only be picked up in location $i \in \mathcal{N}$ if the passenger can walk to their pickup location in time and if their wait time does not exceed the wait time limit, i.e., $t_p^{\text{req}} + \psi_{o(p),i} \leq t$ (pickup time) and $t \leq t_p^{\text{req}} + \psi_{o(p),i} + \Psi$.

Next, we propose a multi-objective formulation of the second-stage problem. Two primary objectives involve maximizing demand coverage and level of service. The latter level-of-service objective is formalized as a three-dimensional objective to reflect the *generalized cost of travel* (Ceder and Wilson, 1986; Desaulniers and Hickman, 2007). Specifically, we model passenger dis-utility via:

1. τ_{rp}^{walk}: walking time from passenger p's origin to the pickup location via subpath $r \in \mathcal{R}_{\ell st}$;
2. τ_{rp}^{wait}: waiting time of passenger p at the pickup location via subpath $r \in \mathcal{R}_{\ell st}$; and
3. $\frac{\tau_{\ell tp}^{delay}}{\tau_p^{direct}}$: relative delay of passenger p at the destination via trip $(\ell, t) \in \mathcal{M}_p$. This metric is normalized with respect to the direct trip time (e.g., a taxi trip). Note that the reference line guarantees that the vehicle reaches the destination at the pre-determined time, so this cost can be expressed at the reference line level as opposed to the subpath level.

We define non-negative hyperparameters λ and μ to weigh the three cost components. Thus, the arc costs in the load-expanded network are formulated as

follows, for all $(\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell,\ s \in \mathcal{S}$:

$$g_a = \begin{cases} \sum_{p \in \mathcal{P}_{r(a)}} D_{ps} \left(\lambda \tau^{walk}_{r(a)p} + \mu \tau^{wait}_{r(a)p} + \frac{\tau^{delay}_{\ell t p}}{\tau^{direct}_{p}} \right) & \forall a \in \bigcup_{r \in \mathcal{R}_{\ell s t}} \mathcal{A}_r, \\ 0 & \forall a \in \mathcal{A}^v_{\ell s t}. \end{cases} \tag{5.6}$$

5.3.3 Two-stage stochastic optimization formulation

The MiND-VRP notation is summarized in Table 5.1. The MiND-VRP maximizes expected ridership and expected level of service. Hyperparameter M tunes the relative weight of these two objectives. The objective function is then given as follows:

$$\min \quad \sum_{s \in \mathcal{S}} \pi_s \left(M \sum_{p \in \mathcal{P}} D_{ps} \Big(1 - \sum_{(\ell,t) \in \mathcal{M}_p} \sum_{a \in \mathcal{A}_{\ell s t} : p \in \mathcal{P}_a} y_a \Big) + \sum_{(\ell,t) \in \mathcal{L} \times \mathcal{T}_\ell} \sum_{a \in \mathcal{A}_{\ell s t}} g_a y_a \right)$$

After omitting the constant term $M \sum_{s \in \mathcal{S}} \pi_s \sum_{p \in \mathcal{P}} D_{ps}$, we formulate the MiND-VRP as follows.

$$\min \quad \sum_{s \in \mathcal{S}} \pi_s \sum_{(\ell,t) \in \mathcal{L} \times \mathcal{T}_\ell} \left(\sum_{a \in \mathcal{A}_{\ell s t}} g_a y_a - M \sum_{p \in \mathcal{P} : (\ell,t) \in \mathcal{M}_p} D_{ps} \sum_{a \in \mathcal{A}_{\ell t s} : p \in \mathcal{P}_{r(a)}} y_a \right) \tag{5.7}$$

$$\text{s.t.} \quad \sum_{\ell \in \mathcal{L} : t \in \mathcal{T}_\ell} h_\ell x_{\ell t} \le B_t \quad \forall t \in \bigcup_{\ell \in \mathcal{L}} \mathcal{T}_\ell \tag{5.8}$$

$$\sum_{(\ell,t) \in \mathcal{M}_p} z_{p \ell t} = 1 \quad \forall p \in \mathcal{P} \tag{5.9}$$

$$\sum_{m:(n,m) \in \mathcal{A}_{\ell s t}} y_{(n,m)} - \sum_{m:(m,n) \in \mathcal{A}_{\ell s t}} y_{(m,n)} = \begin{cases} x_{\ell t} & \text{if } n = u_{\ell s t}, \\ -x_{\ell t} & \text{if } n = v_{\ell s t}, \\ 0 & \text{otherwise}, \end{cases} \tag{5.10}$$
$$\forall (\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell,\ s \in \mathcal{S}, n \in \mathcal{V}_{\ell s t}$$

$$\sum_{a \in \mathcal{A}_{\ell s t} : p \in \mathcal{P}_{r(a)}} y_a \le z_{p \ell t} \quad \forall s \in \mathcal{S}, p \in \mathcal{P}, (\ell, t) \in \mathcal{M}_p \tag{5.11}$$

$$\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z} \text{ binary} \tag{5.12}$$

Equation (5.7) maximizes demand coverage and level of service. First-stage constraints apply a budget B_t (Constraint (5.8)) and ensure that each passenger type is covered by a reference line (Constraint (5.9)). Second-stage constraints enforce flow balance over each load-expanded network (Constraint (5.10)) and ensure consistency between first-stage assignment variables and second-stage passenger service variables (Constraint (5.11)). Equation (5.12) defines the domain of the variables.

Table 5.1: Inputs of the MiND-VRP model.

	Type	**Description**
$\mathcal{L}$	Set	Candidate reference lines
$\mathcal{P}$	Set	Passenger types
$\mathcal{S}$	Set	Demand scenarios
$\mathcal{C}$	Set	Vehicle loads
$\mathcal{I}_\ell$	Set	Reference stops for line $\ell \in \mathcal{L}$, of cardinality I_ℓ
$\mathcal{I}_\ell^{(i)}$	Set	i^{th} stop in reference line $\ell \in \mathcal{L}$ for $i = 1, \cdots, I_\ell$
$\mathcal{T}_\ell$	Set	Allowable departure times of a vehicle from the beginning of line $\ell \in \mathcal{L}$
$\mathcal{M}_p$	Set	Compatible trips in $\mathcal{L} \times \mathcal{T}_\ell$ for passenger type $p \in \mathcal{P}$
$\mathcal{R}_{\ell st}$	Set	Subpaths corresponding to reference trip $(\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell$ in scenario $s \in \mathcal{S}$. Each subpath $r \in \mathcal{R}_{\ell st}$ starts in $u_r \in \mathcal{I}_\ell$ and ends in $v_r \in \mathcal{I}_\ell$.
$(\mathcal{V}_{\ell st}, \mathcal{A}_{\ell st})$	Graph	Load-expanded network of trip $(\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell$ in scenario $s \in \mathcal{S}$. Every trip starts at $u_{\ell st} \in \mathcal{V}_{\ell st}$ and ends at $v_{\ell st} \in \mathcal{V}_{\ell st}$
$\mathcal{A}_r$	Set	Arcs in $\mathcal{A}_{\ell st}$ corresponding to subpath $r \in \mathcal{R}_{\ell st}$ for $(\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell$, $s \in \mathcal{S}$
$\mathcal{A}^v_{\ell st}$	Set	Arcs in $\mathcal{A}_{\ell st}$ connecting line destination to sink node for $(\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell$, $s \in \mathcal{S}$
$\mathcal{P}_r$	Set	Passenger types in $\mathcal{P}$ picked up by subpath $r \in \mathcal{R}_{\ell st}$ for $(\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell$, $s \in \mathcal{S}$
K	Parameter	Number of reference stops a subpath deviation can skip in a row
C	Parameter	Vehicle capacity
B_t	Parameter	Line budget during time unit $t \in \bigcup_{l \in \mathcal{L}} \mathcal{T}_\ell$
h_l	Parameter	Cost to operate one trip via line $\ell \in \mathcal{L}$
D_{ps}	Parameter	Number of passengers of type $p \in \mathcal{P}$ in scenario $s \in \mathcal{S}$
$T_{\ell t}(n)$	Parameter	Time at which trip $(\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell$ must visit stop $n \in \mathcal{I}_\ell$
π_s	Parameter	Probability of scenario $s \in \mathcal{S}$
g_a	Parameter	Cost of arc $a \in \mathcal{A}_{\ell st}$ for trip $(\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell$, scenario $s \in \mathcal{S}$
Ω	Parameter	Maximum walking distance for passengers
Ψ	Parameter	Maximum waiting time for passengers
$\omega_{o,d}$	Parameter	Walking distance between locations o and d
$\psi_{o,d}$	Parameter	Walking time between locations o and d
τ^{walk}_{rp}	Parameter	Walk time of passenger $p \in \mathcal{P}_r$ via subpath $r \in \mathcal{R}_{\ell st}$, for $(\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell$, $s \in \mathcal{S}$
τ^{wait}_{rp}	Parameter	Wait time of passenger $p \in \mathcal{P}_r$ via subpath $r \in \mathcal{R}_{\ell st}$, for $(\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell$, $s \in \mathcal{S}$
$\tau^{delay}_{\ell tp}$	Parameter	Delay of passenger type $p \in \mathcal{P}$ when taking trip $(\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell$
τ^{direct}_p	Parameter	Direct travel time for passenger type $p \in \mathcal{P}$
M	Hyperparameter	Non-negative penalty of unmet demand
λ	Hyperparameter	Non-negative penalty on passenger walk time
μ	Hyperparameter	Non-negative penalty on passenger wait time

5.3.4 Comparison to segment and path-based benchmarks

Recall that the MiND-VRP optimizes subpaths between reference stops in a load-expanded network while ensuring spatial-temporal continuity with flow bal-

ance constraints (Figure 5.1). We compare this formulation to a direct segment-based benchmark and to a path-based benchmark. All formulations are equivalent, but our subpath-based formulation involves fewer variables by implicitly capturing time window constraints without relying on a time-load-expanded network and without involving a full path-based decomposition. Both benchmark models are formulated in Appendix 5.A.

Direct segment-based formulation

This formulation optimizes operations between pickup locations. Each arc corresponds to a road segment connecting stopping locations, as opposed to a subpath that only connects reference stops. A subpath is defined as a sequence of segments that starts at a reference stop at the scheduled time, ends at a subsequent reference stop by the scheduled time, satisfies flow balance in between, and does not skip more than K reference stops.

A vehicle's route impacts the vehicle load as well as the arrival time at the reference stop. To enforce capacity and time window constraints, we could in principle link vehicle routing decisions with load and timing decisions using big-M constraints. To retain a tight second-stage formulation, we instead define a time-load-expanded network, where each node tracks the pickup location, the vehicle load, and the arrival time. The formulation (in Appendix 5.A.1) minimizes the cost function subject to flow balance and passenger assignments (analogous to Equations (5.7)–(5.11)). An extra constraint ensures that the vehicle does not skip more than K stops in a row, which was previously enforced in the definition of subpaths. In summary, the subpath formulation implicitly enforced time window constraints and captured vehicle capacity constraints in a load-expanded network, whereas the segment-based formulation needs to capture both time window and vehicle capacity constraints in a time-load-expanded network.

Path-based formulation

This formulation (in Appendix 5.A.2) optimizes entire paths from the beginning to the end of the reference line. This formulation relates to the subpath-based formulation in that a path is defined as a sequence of subpaths that starts at the beginning of the line, ends at the destination, picks up at most C passengers overall. By construction, a path satisfies the time window constraints at the reference stops and does not skip more than K reference stops in a row. As a result, a path implicitly satisfies the vehicle capacity constraints and all time window constraints. Accordingly, the subpath-based formulation merely minimizes the cost function subject to flow balance and passenger assignments (analogous to Equations (5.7) and (5.11)). An extra constraint ensures that exactly one path is defined for each selected reference line.

Formulation comparison

Proposition 5.1 shows that the three formulations are equivalent.

Proposition 5.1 *The path-based, subpath-based and segment-based formulations achieve the same optimal solution. Moreover, the linear relaxation of the path-based formulation is at least as strong as the one of the subpath-based formulation; and the linear relaxation of the subpath-based formulation is at least as strong as the one of the segment-based formulation.*

5.4 Solution algorithm

The subpath-based formulation of the the MiND-VRP exhibits a two-stage optimization structure with a tight second-stage model. The challenge lies in the size of the model. Accordingly, we propose a solution algorithm that combines Benders decomposition and column generation.

Combinations of column generation and Benders decomposition fall into three categories. One is a simultaneous use of the two methods toward column-and-row generation for problems with column-dependent rows (Muter et al., 2013, 2018). A second one is the use of column generation to solve the Benders master problem (see, e.g., Restrepo et al., 2018; Zeighami and Soumis, 2019). A third one is the use of column generation to solve the Benders subproblem (see, e.g., Mercier et al., 2005; Papadakos, 2009; Karsten et al., 2018). Our approach falls into this third category: Benders decomposition exploits our two-stage stochastic optimization structure (Section 5.4.1), and column generation adds subpaths iteratively in the Benders subproblem (Section 5.4.2). This algorithm relies on a tailored label-setting algorithm to generate subpaths (Section 5.4.2), as well as acceleration strategies (Section 5.4.3). We provide an overview of the algorithm in Section 5.4.4.

Throughout this section, we consider the partial relaxation of the MiND-VRP with first-stage binary variables and second-stage continuous variables, referred to as $\overline{\text{MiND-VRP}}$. This partial relaxation is required because the second-stage problem of the MiND-VRP is not an ideal formulation. Although it primarily relies on flow balance constraints, its constraint matrix is not totally unimodular due to Equation (5.11). Nonetheless, this partial relaxation is close to the full integer optimization problem due to the tight second-stage formulation thanks to the load-expanded network representation. Upon convergence of the Benders decomposition algorithm, we solve a final second-stage model by fixing first-stage solutions and obtaining a feasible MiND-VRP solution after restoring integrality. Throughout the section, we fix a first-stage solution $(\boldsymbol{x}, \boldsymbol{z})$.

5.4.1 Benders reformulation

We propose a multi-cut Benders reformulation of the $\overline{\text{MiND-VRP}}$, which we decompose into a Benders master problem (BMP) and Benders subproblems (BSP). This structure enables a decomposition of the second-stage problem for each reference trip and each demand scenario.

The Benders reformulation of the $\overline{\text{MiND-VRP}}$ includes the first-stage network design and passenger assignment decisions along with a piece-wise linear expression of the recourse function. Note that the subproblem is always feasible: a feasible solution can be constructed by following the reference trip without on-demand deviations, regardless of how many passengers can be served and the consequentially poor level of service. Therefore, the $\overline{\text{MiND-VRP}}$ has relatively complete recourse, and the dual second-stage polyhedron is bounded and nonempty. By the Minkowski-Weyl theorem, we can express the dual second-stage polyhedron as a function of its extreme points.

The second-stage problem can be decomposed into independent subproblems corresponding to each reference trip $(\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell$ and scenario $s \in \mathcal{S}$. Let $\theta_{\ell st}$ denote the corresponding objective:

$$\text{(BSP)} \quad \theta_{\ell st} = \min \quad \sum_{a \in \mathcal{A}_{\ell st}} g_a y_a - M \sum_{p \in \mathcal{P} \,:\, (\ell,t) \in \mathcal{M}_p} D_{ps} \sum_{a \in \mathcal{A}_{\ell st} \,:\, p \in \mathcal{P}_a} y_a \tag{5.13}$$

$$\text{s.t.} \quad \text{Equations (5.10)-(5.11)} \tag{5.14}$$

$$\boldsymbol{y} \geq \mathbf{0} \tag{5.15}$$

Let φ_i and γ_p respectively denote the dual variables corresponding to Equations (5.10) and (5.11). The dual Benders subproblem is then formulated as follows:

$$\max \quad x_{\ell t} \cdot (\varphi_{\bar{u}_{\ell st}} - \varphi_{\bar{v}_{\ell st}}) - \sum_{p \in \mathcal{P} \,:\, (\ell,t) \in \mathcal{M}_p} z_{p\ell t} \cdot \gamma_p \tag{5.16}$$

$$\text{s.t.} \quad \varphi_n - \varphi_m - \sum_{p \in \mathcal{P}_a} \gamma_p \leq g_a - M \sum_{p \in \mathcal{P}_a} D_{ps} \qquad \forall a = (n,m) \in \mathcal{A}_{\ell st} \tag{5.17}$$

$$\varphi_i \in \mathbb{R} \qquad \forall i \in \mathcal{V}_{\ell st} \tag{5.18}$$

$$\gamma_p \geq 0 \qquad \forall p \in \mathcal{P} \,:\, (\ell,t) \in \mathcal{M}_p \tag{5.19}$$

Let $\Lambda_{\ell st}$ denote the set of extreme points of the dual second-stage polyhedron, each corresponding to an optimal second-stage solution $(\boldsymbol{\varphi}, \boldsymbol{\gamma})$ for a given reference trip $(\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell$ and scenario $s \in \mathcal{S}$. Let $\boldsymbol{\Lambda} = (\Lambda_{\ell st})_{(\ell,t) \in \mathcal{L} \times \mathcal{T}_\ell, s \in \mathcal{S}}$ denote the full set of extreme points across reference trips and across scenarios. The

multi-cut Benders reformulation of the $\overline{\text{MiND-VRP}}$ is then given as:

$$\text{BMP}(\boldsymbol{\Lambda}) \quad \min \quad \sum_{s\in\mathcal{S}} \pi_s \Big(M \sum_{p\in\mathcal{P}} D_{ps} + \sum_{\ell\in\mathcal{L}} \sum_{t\in\mathcal{T}_\ell} \theta_{\ell st} \Big) \tag{5.20}$$

$$\text{s.t.} \quad \text{Equations (5.8)–(5.9)} \tag{5.21}$$

$$\theta_{\ell st} \geq x_{\ell t} \cdot (\varphi_{\bar{u}_{\ell st}} - \varphi_{\bar{v}_{\ell st}}) - \sum_{p\in\mathcal{P}\,:\,(\ell,t)\in\mathcal{M}_p} z_{p\ell t} \cdot \gamma_p, \quad \forall (\ell,t) \in \mathcal{L}\times\mathcal{T}_\ell,\ s\in\mathcal{S},\ (\boldsymbol{\varphi},\boldsymbol{\gamma}) \in \Lambda_{\ell st} \tag{5.22}$$

$$\boldsymbol{x}, \boldsymbol{z} \text{ binary} \tag{5.23}$$

To circumvent the exponential number of extreme points, Benders decomposition proceeds via row generation. The Benders master problem solves a relaxation containing a subset of constraints $\overline{\boldsymbol{\Lambda}} \subseteq \boldsymbol{\Lambda}$, amounting to a lower-bounding approximation of the recourse function. It is given by solving BMP($\overline{\boldsymbol{\Lambda}}$) instead of BMP($\boldsymbol{\Lambda}$).

By design, the BMP yields a lower bound of the $\overline{\text{MiND-VRP}}$ and the combination of the BMP and BSP yield an upper bound of the $\overline{\text{MiND-VRP}}$. If the optimality gap lies within a given tolerance, the algorithm stops with a provably optimal solution of the $\overline{\text{MiND-VRP}}$. Otherwise, we retrieve the optimal dual solution of the BSP $(\boldsymbol{\varphi},\boldsymbol{\gamma})$, and add the following optimality cut to the BMP.

$$\theta_{\ell st} \geq x_{\ell t} \cdot (\bar{\varphi}_{\bar{u}_{\ell st}} - \bar{\varphi}_{\bar{v}_{\ell st}}) - \sum_{p\in\mathcal{P}\,:\,(\ell,t)\in\mathcal{M}_p} z_{p\ell t} \cdot \bar{\gamma}_p \tag{5.24}$$

By iterating between the BMP and BSP, the Benders decomposition algorithm converges to an optimal solution of the $\overline{\text{MiND-VRP}}$. However, the subproblem involves a large number of subpath-based variables, which can slow down the Benders decomposition scheme. In response, we leverage column generation to add subpaths iteratively in the Benders subproblem.

5.4.2 Column generation procedure for Benders subproblem

We focus in this section on the BSP (Equations (5.13)–(5.15)) for a given reference trip $(\ell,t) \in \mathcal{L}\times\mathcal{T}_\ell$ and scenario $s \in \mathcal{S}$. The set of arcs $\mathcal{A}_{\ell st}$ grows exponentially with the number of possible pickup locations between reference stops, especially as microtransit vehicles are allowed to skip reference stops via longer subpaths ($K > 0$). Thus, we propose a column generation procedure that dynamically generates subpaths. We decompose the BSP into a restricted master problem (RMP) based on a subset of subpaths and a pricing problem (PP) that iteratively generates subpaths. The main feature of our microtransit problem is that it iteratively generates subpaths between reference stops as

opposed to full paths from start to finish, as in traditional path-based column generation. We present our subpath-based column generation approach below, with a particular focus on the structure of subpaths and the implications for the pricing problem.

Restricted master problem

The RMP simply solves the Benders subproblem with a subset of subpath-based arcs by $\mathcal{A}'_{\ell st} \subseteq \mathcal{A}_{\ell st}$. The RMP is then formulated as follows:

$$\text{RMP}(\mathcal{A}'_{\ell st}) \quad \min \quad \sum_{a \in \mathcal{A}'_{\ell st}} g_a y_a - M \sum_{p \in \mathcal{P} \,:\, (\ell,t) \in \mathcal{M}_p} D_{ps} \sum_{a \in \mathcal{A}'_{\ell st} \,:\, p \in \mathcal{P}_a} y_a \tag{5.25}$$

s.t.

$$\sum_{m:(n,m) \in \mathcal{A}'_{\ell st}} y_{(n,m)} - \sum_{m:(m,n) \in \mathcal{A}'_{\ell st}} y_{(m,n)} = \begin{cases} x_{\ell t} & \text{if } n = \bar{u}_{\ell st}, \\ -x_{\ell t} & \text{if } n = \bar{v}_{\ell st}, \\ 0 & \text{otherwise,} \end{cases} \quad \forall n \in \mathcal{V}_{\ell st} \tag{5.26}$$

$$\sum_{a \in \mathcal{A}'_{\ell st} \,:\, p \in \mathcal{P}_{r(a)}} y_a \leq z_{p\ell t} \quad \forall p \in \mathcal{P} : (\ell, t) \in \mathcal{M}_p \tag{5.27}$$

$$\boldsymbol{y} \geq \boldsymbol{0} \tag{5.28}$$

Subpath characterization

Toward a column generation pricing problem, we optimize the intermediate stopping locations of a candidate subpath $r \in \mathcal{R}_{\ell st}$. For expositional clarity, we denote the source and sink stops as $u = u_r$ and $v = v_r$. Let $\mathcal{N}_{\ell uv}$ denote the set of all pickup locations between u and v; and let $\mathcal{E}_{\ell uv} \subseteq \mathcal{N}_{\ell uv} \times \mathcal{N}_{\ell uv}$ denote the set of road segments connecting those stopping locations. Each subpath will be characterized by a sequence of segments in $\mathcal{E}_{\ell uv}$ satisfying the capacity, time window, and stop-skipping requirements outlined in Section 5.3.2. Note that the Benders subproblem, hence the column generation restricted master problem, combine subpaths from the beginning to the end of a reference line, whereas the column generation pricing problem creates subpaths by combining road segments between reference stops.

To efficiently capture time window constraints in the pricing problem, we characterize subpaths in a time-expanded network, denoted by $(\mathcal{U}^{uv}_{\ell st}, \mathcal{H}^{uv}_{\ell st})$. Let $\mathcal{T}^{uv}_{\ell t}$ be the set of discretized time intervals between the scheduled departure time from reference stop u and the scheduled arrival time at reference stop v, i.e., between $\mathcal{T}_{\ell t}(u)$ and $\mathcal{T}_{\ell t}(v)$. Each node $m \in \mathcal{U}^{uv}_{\ell st}$ is represented by a tuple (k_m, t_m) where $k_m \in \mathcal{N}_{\ell uv}$ is a stopping location between reference stops u and v, and

$t_m \in \mathcal{T}_{\ell t}^{uv}$ is the time at which the vehicle arrives at stop k_m. The subpath's source is $(u, T_{\ell t}(u)) \in \mathcal{U}_{\ell st}^{uv}$, and its sink is $(v, T_{\ell t}(v)) \in \mathcal{U}_{\ell st}^{uv}$. The arc set $\mathcal{H}_{\ell st}^{uv}$ comprises traveling arcs and idling arcs. Traveling arcs connect any node pair $(i,t) \to (j, t+tt_{ij})$ where $(i,j) \in \mathcal{E}_{\ell uv}$ defines a road segment and tt_{ij} defines the corresponding travel time, with $t \in \mathcal{T}_{\ell t}^{uv}$ and $t + tt_{ij} \in \mathcal{T}_{\ell t}^{uv}$. Idling arcs connect any node pair $(i,t) \to (i, t+1)$ where $i \in \mathcal{N}_{\ell uv}$ defines a stopping location and $t \in \mathcal{T}_{\ell t}^{uv}$ defines the stopping time. Idling arcs are required to capture instances where the vehicle waits for a passenger at a pickup location or waits at the reference stop to adhere to the reference schedule.

Each node also tracks the passengers that can be served and their corresponding waiting time, walking time, and arrival delay. We let $\mathcal{P}_m$ denote the set of passengers that can be picked up at node $m \in \mathcal{U}_{\ell st}^{uv}$, i.e. the set of passenger requests with $D_{ps} > 0$, where node m satisfies the walking and waiting restrictions defined in Section 5.3.2. The notation is summarized in Table 5.2 and the level of service is given by:

$$D_{ps}\left(\frac{\tau_{\ell tp}^{delay}}{\tau_p^{direct}} + \lambda \tau_{mp}^{\text{walk}} + \mu \tau_{mp}^{\text{wait}}\right) \tag{5.29}$$

Table 5.2: Time-expanded network components.

	Type	**Description**
$\mathcal{N}_{\ell uv}$	Set	Nodes of possible stopping locations between reference stops u and v
$\mathcal{E}_{\ell uv}$	Set	Directed arcs in $\mathcal{N}_{\ell uv} \times \mathcal{N}_{\ell uv}$, corresponding to road segments
$(\mathcal{U}_{\ell st}^{uv}, \mathcal{H}_{\ell st}^{uv})$	Graph	Time-expanded network from $(u, T_{\ell t}(u))$ to $(v, T_{\ell t}(v))$. Node $m \in \mathcal{U}_{\ell st}^{uv}$ is characterized by a location-time tuple (k_m, t_m)
$\mathcal{T}_{\ell t}^{uv}$	Set	Time intervals between the scheduled times $T_{\ell t}(u)$ and $T_{\ell t}(v)$
$\mathcal{P}_m$	Set	Passengers in $\mathcal{P}$ that can be picked up in node $m \in \mathcal{U}_{\ell st}^{uv}$
tt_e	Parameter	Travel time corresponding to road segment $e \in \mathcal{E}_{\ell uv}$
τ_{mp}^{walk}	Parameter	Walk time of passenger $p \in \mathcal{P}_m$ when picked up at node $m \in \mathcal{U}_{\ell st}^{uv}$
τ_{mp}^{wait}	Parameter	Wait time of passenger $p \in \mathcal{P}_m$ when picked up at node $m \in \mathcal{U}_{\ell st}^{uv}$

Pricing problem

We define the following decision variables to build a subpath:

$$f_{mq} = \begin{cases} 1 & \text{if arc } (m,q) \in \mathcal{H}_{\ell st}^{uv} \text{ is traversed,} \\ 0 & \text{otherwise.} \end{cases}$$

$$w_{mp} = \begin{cases} 1 & \text{if passenger } p \in \mathcal{P}_m \text{ is picked up at node } m \in \mathcal{U}^{uv}_{\ell st}, \\ 0 & \text{otherwise.} \end{cases}$$

Recall that the BSP optimizes over arc variables in the load-expanded network. Therefore, each generated column must correspond to a sequence of road segments as well as a passenger pickup set. Specifically, let us consider a BSP variable $a = (start(a), end(a)) \in \mathcal{A}_{\ell st}$. This variable corresponds to subpath $r(a) \in \mathcal{R}_{\ell st}$ from reference stop $u_{r(a)}$ to reference stop $v_{r(a)}$. In addition, the subpath must satisfy the incremental load constraint $\sum_{p \in P_{r(a)}} D_{ps} = c_{end(a)} - c_{start(a)}$. From Equations (5.17) and (5.29), the reduced cost $\widehat{g}_a$ consists of a subpath component and a load component:

$$\widehat{g}_a = \underbrace{\sum_{m \in \mathcal{U}^{uv}_{\ell st}} \sum_{p \in \mathcal{P}_m} d_{mp} w_{mp}}_{\text{subpath component}} \underbrace{-\varphi_{start(a)} + \varphi_{end(a)}}_{\text{load component}}$$

$$\forall a \in \mathcal{A}_{\ell st}, \text{ with } u = u_{r(a)},\ v = v_{r(a)}, \quad (5.30)$$

$$\text{and } d_{mp} = D_{ps} \left(\frac{\tau^{delay}_{\ell tp}}{\tau^{direct}_{p}} + \lambda \tau^{\text{walk}}_{mp} + \mu \tau^{\text{wait}}_{mp} \right) - (\gamma_p + M D_{ps}),$$

$$\forall m \in \mathcal{U}^{uv}_{\ell st}, p \in \mathcal{P}_m. \quad (5.31)$$

This reduced cost can be interpreted as the combination of: (i) the level-of-service penalty for passengers receiving a service; (ii) the value of serving a passenger, captured by the actual value M and the dual price γ_p; and (iii) the cost of increasing the vehicle load, reflected in the dual prices $\varphi_{start(a)}$ and $\varphi_{end(a)}$. We now formulate the following pricing problem, which seeks the variable with the most negative reduced cost connecting nodes $start(a) \in \mathcal{V}_{\ell st}$ and $end(a) \in \mathcal{V}_{\ell st}$ corresponding to reference stops $k_{start(a)} = u \in \mathcal{N}$ and $k_{end(a)} = v \in \mathcal{N}$ and loads $c_{end(a)} \geq c_{start(a)}$, for reference trip $(\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell$ and scenario $s \in \mathcal{S}$. We denote by $Z^a_{\ell st}$ its optimal objective value.

$$\text{(PP)} \quad \min \quad \sum_{m \in \mathcal{U}^{uv}_{\ell st}} \sum_{p \in \mathcal{P}_m} d_{mp} w_{mp} - \varphi_{start(a)} + \varphi_{end(a)} \quad (5.32)$$

$$\text{s.t.} \quad \sum_{p \in \mathcal{P}_m} w_{mp} \leq \sum_{q:(m,q) \in \mathcal{H}^{uv}_{\ell st}} f_{mq} \quad \forall m \in \mathcal{U}^{uv}_{\ell st} \quad (5.33)$$

$$\sum_{u \in \mathcal{U}^{uv}_{\ell st}} \sum_{p \in \mathcal{P}_m} D_{ps} w_{mp} = c_{end(a)} - c_{start(a)} \quad (5.34)$$

$$\sum_{m \in \mathcal{U}^{uv}_{\ell st} : p \in \mathcal{P}_u} w_{mp} \leq 1 \quad \forall p \in \mathcal{P} \quad (5.35)$$

$$\sum_{q:(m,q)\in\mathcal{H}^{uv}_{\ell st}} f_{mq} - \sum_{q:(q,m)\in\mathcal{H}^{uv}_{\ell st}} f_{qm} = \begin{cases} 1 & \text{if } m = (u, T_{\ell t}(u)), \\ -1 & \text{if } m = (v, T_{\ell t}(v)), \\ 0 & \text{otherwise.} \end{cases} \quad \forall m \in \mathcal{U}^{uv}_{\ell st} \tag{5.36}$$

$$f_{mq} \in \{0,1\} \quad \forall (m,q) \in \mathcal{H}^{uv}_{\ell st} \tag{5.37}$$

$$w_{mp} \in \{0,1\} \quad \forall m \in \mathcal{U}^{uv}_{\ell st}, p \in \mathcal{P}_m \tag{5.38}$$

This formulation minimizes the reduced cost of a subpath-based variable in the load-expanded network corresponding to arc $a \in \mathcal{S}_{\ell st}$ (Equation (5.32)). Constraints (5.33) enforce the consistency between variables, by stating that a passenger can only be picked up if the corresponding node is visited. Constraints (5.34) define the load difference between node n and m as the number of passengers served. Constraints (5.35) guarantee that a passenger is picked up at most once. Constraints (5.36) apply flow balance. Equations (5.37) and (5.38) define the domain of the variables.

Note that depending on the passenger set $\mathcal{P}$, an arc $(start(a), end(a))$ may not exist if the load deviation is higher than the number of passengers that can be served between the reference stops within a feasible deviation. To prevent infeasibility, one can precompute the maximum number of passengers that can be picked up between reference stops, and only consider nodes $(start(a), end(a))$ that at most serve that number of passengers. Despite that, the number of pricing problems can be significantly large. We can reduce the number of pricing problems, while still guaranteeing column generation to converge, by computing φ^{max}_{cuv}, for each load differential $\Delta c \in \mathcal{C}$, defined as the maximum value of $\varphi_{start(a)} - \varphi_{end(a)}$ where $c_{end(a)} - c_{start(a)} = \Delta c$. Then we can solve a PP that seeks a subpath of c passengers, and with only the subpath-based component in the objective. Let $Z^{cuv}_{\ell st}$ be the optimal solution of this problem. The value $Z^{cuv}_{\ell st} - \varphi^{max}_{cuv}$ corresponds to the arc variable of minimum reduced cost for a subpath with load differential Δc. This procedure reduces the number of pricing problem, while retaining the exactness of the column generation approach.

Proposition 5.2 *At most one pricing problem for each load differential $\Delta c \in \mathcal{C}$ is sufficient to find the arc variable with the minimum reduced cost.*

Column generation iterates between the restricted master problem and the pricing problem. If all reduced costs are non-negative, then the column generation algorithm terminates and the Benders decomposition algorithm proceeds. Otherwise, the pricing problem identifies a subpath-based arc $a \in \mathcal{A}_{\ell st}$ with negative reduced cost and adds it to the load-expanded network, by augmenting

$\mathcal{A}'_{\ell st} \leftarrow \mathcal{A}'_{\ell st} \cup \{a\}$, and defining g_a as the following level of service:

$$g_a = \sum_{m \in \mathcal{U}^{uv}_{\ell st}} \sum_{p \in \mathcal{P}_m} D_{ps} \left(\frac{\tau^{delay}_{\ell tp}}{\tau^{direct}_{p}} + \lambda \tau^{\text{walk}}_{mp} + \mu \tau^{\text{wait}}_{mp} \right) w_{mp} \tag{5.39}$$

Label setting algorithm

Due the structure of the subpaths and the decomposability of the reduced cost, the pricing problem exhibits a resource-constrained shortest path structure. We design a label-setting algorithm by exploiting the directed and acyclic structure of $(\mathcal{U}^{uv}_{\ell st}, \mathcal{H}^{uv}_{\ell st})$ (Ahuja et al., 1993). The label stores the set of served passengers and the corresponding service level. By keeping track of all non-dominated labels, the algorithm identifies all subpath-based variables with negative reduced cost.

State definition. Let $(m^\sigma, \mathbb{P}^\sigma)$ denote a state, where m^σ tracks the "current" node, and $\mathbb{P}^\sigma$ tracks the set of served passengers $p \in \mathcal{P}$ with pickup node ρ_p. We keep track of the reduced cost $G(m^\sigma, \mathbb{P}^\sigma)$.

Initial state: $(m^0 = m : k_m = u, t_m = T_{\ell t}(u), \mathbb{P}^0 = \emptyset)$, with $G(m^0, P^0) = 0$.

State transitions. For each arc $(m, q) \in \mathcal{H}^{uv}_{\ell st}$ traversed, and all passenger combinations $\mathbb{P}_m \subseteq \mathcal{P}_m$, the state is updated to $(q, \mathbb{P}^\sigma \cup \mathbb{P}_m)$. For each new passenger $p \in \mathbb{P}_m \setminus \{\mathbb{P}^\sigma\}$, the pickup point is set to $\rho_p = m$. For existing passengers $p \in \mathbb{P}_m \cap \mathbb{P}^\sigma$, we update the pickup node to be $\rho_p = m$ if m provides a better service level—that is, if $d_{mp} < d_{\rho_p, p}$. This transition is admissible if the vehicle has sufficient capacity, i.e., if $\sum_{p \in \mathbb{P}^\sigma} D_{ps} + \sum_{p \in \mathbb{P}_m \setminus \{\mathbb{P}^\sigma\}} D_{ps} \leq C$.

Reward function. $G(m^\sigma, \mathbb{P}^\sigma) = \sum_{p \in \mathbb{P}^\sigma} d_{\rho_p, p}$ tracks the reduced cost of a subpath up to state σ.

Dominance rule. State σ^1 dominates state σ^2 if (i) $m^{\sigma^1} = m^{\sigma^2}$, (ii) $\mathbb{P}^{\sigma^1} = \mathbb{P}^{\sigma^2}$, and (iii) $G(m^{\sigma^1}, \mathbb{P}^{\sigma^1}) \leq G(m^{\sigma^2}, \mathbb{P}^{\sigma^2})$. Upon termination, we extract all non-dominated states l such that $m^\sigma = m : k_m = v$ and $t_m = T_{\ell t}(v)$, and we define a subpath $r \in \mathcal{R}_{\ell st}$ where $\mathcal{P}_r = \mathbb{P}^\sigma$.

Note that the dominance rule is rather weak, leading to the enumeration of different subpaths with different passenger combinations. Still, this dominance rule is required for the label-setting algorithm to find the subpath with the lowest reduced cost for any load differential, thus providing a certificate of optimality in the column generation algorithm.

Upon termination, the label-setting algorithm yields a subpath $r \in \mathcal{R}_{\ell st}$. We

then identify all arcs $a \in \mathcal{A}_{\ell st} \setminus \{\mathcal{A}'_{\ell st}\}$ such that we can add to the RMP, that is, all arcs such that $k_{start(a)} = u,\ k_{end(a)} = v,\ c_{end(a)} - c_{start(a)} = \sum_{p \in \mathbb{P}^\sigma} D_{ps}$. We compute their reduced cost as follows:

$$\hat{g}_a = G(m^\sigma, \mathbb{P}^\sigma) - \varphi_{start(a)} + \varphi_{end(a)} \tag{5.40}$$

The main advantage of the label-setting algorithm over directly solving the pricing problem is that we only need to apply it once between pair of reference stops u and v instead of $\mathcal{O}(C)$ times (one for each load differential). The dominance rule ensures that all subpaths with different passenger combinations are computed, therefore including the minimum reduced cost subpath for each possible load differential. This also allows us to identify not only the most negative reduced cost but to add multiple columns with negative reduced cost in the same iteration.

5.4.3 Acceleration strategies

We have implemented several acceleration strategies to speed up the algorithm. Two techniques were particularly effective and resulted in significant improvement in model scalability—one relating to the Benders decomposition procedure and one relating to the column generation procedure.

Managing the search tree

We design a two-phase procedure to add Benders cuts at the root node of the search tree. In the first phase, we apply Benders decomposition to the full relaxation of the MiND-VRP, with continuous first-stage and second-stage decisions. At the end of the first phase, we retain all the Benders cuts and restore integrality in the BMP. In the second phase, we then proceed to applying Benders decomposition to the partial relaxation $\overline{\text{MiND-VRP}}$.

This strategy can significantly improve the algorithm's performance (Rahmaniani et al., 2018). It relies on a more efficient Benders master problem in initial iterations, which accelerates the generation of Benders cuts and warm starts the second phase with a stronger recourse function approximation. Moreover, the second phase starts with a tighter problem formulation, which can help reduce the branch-and-bound tree search in the BMP at each iteration of the algorithm.

Column filtering

The label-setting algorithm keeps track of two components: which passengers to pick up and at which node they should be picked up. By design, the microtransit system brings flexibility regarding where to pick up a passenger. From a practical standpoint, such flexibility can enhance the efficiency of on-demand

operations. From a technical standpoint however, it weakens the dominance criteria in the label-setting algorithm, which can lead to extensive enumeration.

To circumvent this challenge, we present a heuristic acceleration to discard more labels at each step. Specifically, when visiting a node $m \in \mathcal{U}_{\ell st}^{ij}$, we always serve a candidate passenger $p \in \mathcal{P}_m$ as long as the vehicle does not operate at capacity and as long as the corresponding reduced cost is negative (i.e., $d_{mp} < 0$). In practice, this approach is motivated by the fact that subpaths are relatively short, hence it is unlikely that the pricing problem would deliberately decide to not serve a passenger in order to free up capacity for a subsequent passenger. Moreover, it allevates the undesirable situation of a vehicle visiting a stop to pick up some passengers and rejecting others. On the negative side, this strategy can be less effective in the presence of high-demand points concentrated in time and space, in which case it can exclude promising variables.

We note that this heuristic corresponds to an upper-bounding approximation of the pricing problem. Therefore, any solution generated via this heuristic does define a valid subpath with negative reduced cost. However, the heuristic could potentially find no subpath with negative reduced cost even though the column generation algorithm has not converged to an optimal solution. In that case, we can switch back to the full label-setting algorithm in final iterations in order to derive a certificate of optimality. In our experiments, we have found that this heuristic results is significant speedups in the subproblem without compromising solution quality.

5.4.4 Summary of solution algorithm

Our solution algorithm, summarized in Figure 5.2, involves two interconnected decomposition structures to solve the partial relaxation $\overline{\text{MiND-VRP}}$. An outer loop solves the problem via Benders decomposition, by iterating between the BMP, which generates a feasible first-stage solution and a valid lower bound, and the BSP, which yields a valid upper bound. At each outer iteration, the algorithm provides a certificate of optimality, or otherwise generates an optimality cuts for the BMP to guide the algorithm toward convergence. Then, an inner loop solves the BSP using column generation. Starting with a subset of subpath-based variables, the procedure iterates between the RMP, which generates a feasible solution to the BSP, and the PP, which provides a certificate of optimality or otherwise identifies new subpath-based variables with negative reduced cost. These columns are added to the RMP and the process continues until inner convergence. This approach relies on a tailored two-label dynamic programming algorithm to solve the PP effectively.

Upon convergence, the algorithm provides a feasible solution to the partial relaxation $\overline{\text{MiND-VRP}}$; however, the second-stage variables may still be fractional.

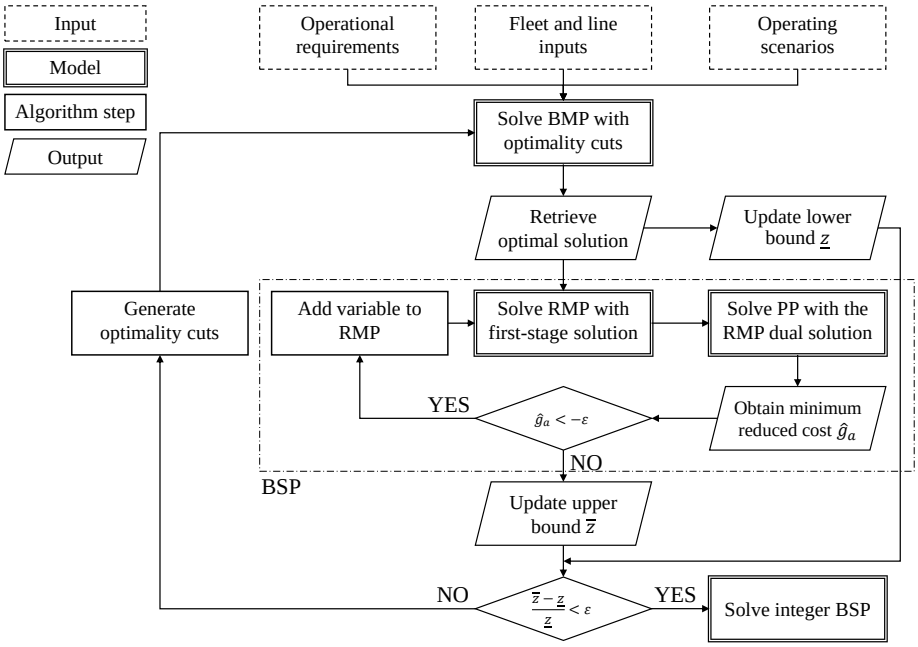


Figure 5.2: Algorithm overview.

Therefore, we solve the second-stage problem one last time in order to retrieve a feasible integer solution to the MiND-VRP. The optimal linear solution is still a valid lower bound to the problem, which generates a valid optimality gap. As we shall see experimentally, the optimality gap is very small due to the tight second-stage formulation in the MiND-VRP.

Proposition 5.3 *The solution algorithm returns an optimal solution to the partial relaxation $\overline{\text{MiND-VRP}}$ in a finite number of iterations.*

Note, finally, that our label-setting algorithm can also be used offline to enumerate the entire set $\mathcal{A}_{\ell st}$ for each reference trip $(\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell$ and each scenario $s \in \mathcal{S}$, upon excluding the dual values from the cost function. Therefore, our algorithmic tools also enable to solve the full MiND-VRP via integer optimization or its partial relaxation $\overline{\text{MiND-VRP}}$ with Benders decomposition alone.

5.5 Computational results

We evaluate our modeling and algorithmic approach using the taxi data from the (NYC Taxi & Limousine Commission, 2021). Since the MiND-VRP for-

mulation assumes all passengers have a common destination, we define a case study corresponding to an airport shuttle service. We filter the trips during the morning rush (6-9am) traveling from Manhattan to LaGuardia airport.

We divide Manhattan into a set of 1570 potential pickup locations with even coverage. We use data from the Metropolitan Transportation Authority (2022) to define 38 candidate lines, as detailed in Appendix 5.B.2. There are an average of 11.2 reference stops per line, ranging from 3 to 21 reference stops. Figure 5.3 illustrates this experimental setup.

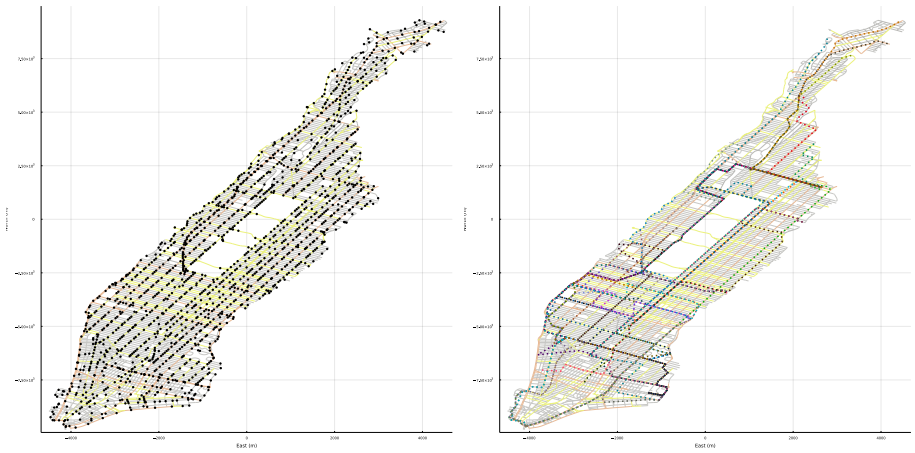


Figure 5.3: Network setup. *Left:* Set of pickup locations. *Right:* Set of candidate reference lines.

The size of the MiND-VRP is governed by the number of candidate reference lines (ranging from 5 to 38 in out experiments), the planning horizon (ranging from 30 minutes to three hours), the parameter K defining the number of reference stops that subpaths can skip (set to 0 or 1), and the number of demand scenarios (ranging from 5 to 50). On average, our problem takes 1,118 passenger requests as inputs with the full three-hour planning horizon, with a range from 762 requests to 1,659 requests across the 50 scenarios. We consider a homogeneous fleet of vehicles, with a capacity of 10 passengers. We assume that transit vehicles can be scheduled every 15 minutes; we aggregate demand across three-minute intervals in the Benders master problem, and discretize time in 30-second increments in the pricing problem. Given these specifications, the largest instances contain over 2.5 million first-stage variables, 25,000 Benders sub-problems and 500,000 pricing problems.

All models are solved with Gurobi v8.1, using the optimization package JuMP

in Julia (Dunning et al., 2017). We impose a three-hour time limit for the algorithms and a ten-hour total limit with pre-processing. We list all additional parameters in Appendix 5.B.1. The remainder of this section evaluates the benefits of our methodology toward solving large-scale instances of the MiND-VRP arising in practice, and the impact of demand-responsive microtransit on urban mobility.

5.5.1 Benefits of subpath-based model formulation

Table 5.3 compares our subpath-based formulationto the path-based and segment-based benchmarks described in Section 5.3.4. All models are solved using off-the-shelf mixed-integer optimization solvers. We consider a value of $K = 0$ here, meaning that microtransit vehicles must visit all reference stops.

Note, first, that the segment-based formulation does not scale to even small instances, due to the extended formulation in a time-load-expanded network that is required to handle the capacity and time window constraints in the second stage. As an indication, a model with two lines, 15 minutes of demand horizon, and a single scenario contains more than 16 million variables in the segment-based formulation. In the path-based formulation, the total number of paths can grow exceedingly large. To ensure a fair comparison, we limit the number of paths to one million for each sub-problem. The results highlight the value of subpath-based modeling: the subpath-based model scales significantly better than the path-based models, both in terms of problem size and solution time. In small-scale instances, the subpath-based model terminates up to 20 times faster than the path-based formulation. Moreover, the subpath-based model is solved to optimality in larger instances where the path-based model runs out of memory.

Yet, all models fail to scale to the largest instances, for instances with a three-hour horizon and 50 scenarios. In addition, none of the instances with $K = 1$ could be solved, due to the intractable size of the second-stage variable set. This motivates the need for decomposition algorithms.

5.5.2 Benefits of decomposition algorithm

Table 5.4 compares Benders decomposition alone, our exact algorithm combining Benders decomposition and column generation, and our accelerated algorithm with the heuristic column generation procedure. For each method, we report the optimality gap *Gap*, the solution time *CPU*, and the number of subpaths generated $|\mathcal{A}_{\ell st}|$.

Table 5.3: Performance comparison between path-based, subpath-based and segment-based formulations."—" indicates that the memory limit was reached. $|\mathcal{Q}_{\ell st}|$ and $|\mathcal{A}_{\ell st}|$ in thousands, computational times in seconds.

			Path				**Subpath**				**Segment**
$\|\mathcal{L}\|$	Horizon	$\|\mathcal{S}\|$	Pre-proc.	Solve	Total	$\|\mathcal{Q}_{\ell st}\|$	Pre-proc.	Solve	Total	$\|\mathcal{A}_{\ell st}\|$	
5	30 min.	5	99	1	100	232	50	1	51	41	—
		10	296	2	298	767	57	2	59	82	—
		25	1,473	8	1,481	3,156	164	3	167	211	—
		50	1,804	13	1,817	4,889	498	5	503	426	—
	90 min.	5	1,585	18	1,603	5,567	380	5	385	94	—
		10	2,597	29	2,626	9,611	457	7	464	188	—
		25	13,566	108	13,674	31,251	2,306	12	2,318	512	—
		50	—	—	—	—	4,058	20	4,078	1,078	—
	180 min.	5	2,946	36	2,982	9,386	549	16	565	170	—
		10	4,931	81	5,012	16,323	964	23	987	341	—
		25	—	—	—	—	4,546	29	4,575	890	—
		50	—	—	—	—	9,516	44	9,560	1,896	—
10	30 min.	5	159	3	162	251	100	3	103	81	—
		10	517	4	521	804	154	4	158	163	—
		25	2,518	16	2,534	5,389	516	7	523	424	—
		50	5,477	41	5,518	10,538	1,522	14	1,536	879	—
	90 min.	5	2,990	38	3,028	8,396	556	13	569	186	—
		10	5,082	65	5,147	17,612	835	15	850	399	—
		25	—	—	—	—	3,773	31	3,804	1,078	—
		50	—	—	—	—	7,224	160	7,384	2,294	—
	180 min.	5	5,374	87	5,461	15,007	1,119	38	1,157	344	—
		10	—	—	—	—	2,324	43	2,367	715	—
		25	—	—	—	—	8,754	75	8,829	1,873	—
		50	—	—	—	—	—	—	—	—	
20	30 min.	5	882	6	888	288	773	7	780	150	—
		10	1,595	10	1,605	1,707	1,053	9	1,062	315	—
		25	—	—	—	—	—	—	—	—	—
		50	—	—	—	—	—	—	—	—	—
38	30 min.	5	1,791	14	1,805	503	1,704	15	1,719	263	—
		10	3,513	24	3,537	3,797	2,359	18	2,377	559	—
		25	—	—	—	—	—	—	—	—	—
		50	—	—	—	—	—	—	—	—	—

Table 5.4: Comparison of Benders + column enumeration (BCE), Benders + column generation (BCG), and Benders + heuristic column generation (BHCG). "*": time limit exceeded. "—": memory limit exceeded. "NA": gap unavailable for BHCG because BCG ran out of memory. $|\mathcal{A}_{\ell st}|$ in thousands.

			$K=0$									$K=1$								
			BCE			**BCG**			**BHCG**			**BCE**			**BCG**			**BHCG**		
$\|\mathcal{L}\|$	Horizon	$\|\mathcal{S}\|$	Gap	CPU(s)	$\|\mathcal{A}_{\ell st}\|$	Gap	CPU(s)	$\|\mathcal{A}_{\ell st}\|$	Gap	CPU(s)	$\|\mathcal{A}_{\ell st}\|$	Gap	CPU(s)	$\|\mathcal{A}_{\ell st}\|$	Gap	CPU(s)	$\|\mathcal{A}_{\ell st}\|$	Gap	CPU(s)	$\|\mathcal{A}_{\ell st}\|$
5	30	5	0.0	46	41	0.0	92	32	0.0	80	31	—	—	—	0.0	689	78	1.0	335	67
		10	0.0	37	82	0.0	138	63	0.1	105	63	—	—	—	0.0	1,854	158	0.7	476	136
		25	0.0	72	211	0.0	368	160	0.1	280	158	—	—	—	1.3	*	471	1.3	1,606	350
		50	0.0	118	426	0.0	697	323	0.1	460	320	—	—	—	—	—	—	NA	3,764	712
	90	5	0.0	188	94	0.0	315	68	0.0	238	67	—	—	—	0.0	3,649	189	0.6	1,074	151
		10	0.0	187	188	0.0	397	137	0.0	386	135	—	—	—	0.0	*	409	0.3	1,966	311
		25	0.0	326	512	0.0	1,024	350	0.0	809	342	—	—	—	—	—	—	NA	4,988	804
		50	0.0	510	1,078	0.0	2,240	711	0.1	1,697	696	—	—	—	—	—	—	—	—	—
	180	5	0.0	539	170	0.0	747	120	0.0	661	118	—	—	—	0.0	8,097	330	0.5	2,514	270
		10	0.0	629	341	0.0	1,010	242	0.1	913	238	—	—	—	20.2	*	760	17.3	5,034	551
		25	0.0	903	890	0.0	2,681	614	0.1	1,928	601	—	—	—	—	—	—	NA	*	1,410
		50	—	—	—	0.0	5,895	1,251	0.1	5,773	1,224	—	—	—	—	—	—	—	—	—
10	30	5	0.0	114	81	0.0	305	66	0.0	201	65	—	—	—	0.0	2,152	153	1.5	822	139
		10	0.0	146	163	0.0	332	132	0.0	378	131	—	—	—	0.0	5,341	309	1.2	1,817	281
		25	0.0	231	424	0.0	741	334	0.2	838	331	—	—	—	28.4	*	874	16.0	5,985	723
		50	0.0	419	879	0.0	1,870	676	0.2	1,513	669	—	—	—	—	—	—	NA	*	1,466
	90	5	0.0	526	186	0.0	621	140	0.1	636	139	—	—	—	0.4	*	356	1.1	3,745	309
		10	0.0	644	399	0.0	1,174	283	0.1	1,054	281	—	—	—	8.1	*	761	4.0	9,960	636
		25	0.0	996	1,078	0.0	3,018	724	0.2	3,087	712	—	—	—	—	—	—	—	—	—
		50	0.0	2,011	2,294	0.0	7,545	1,472	0.2	16,695	1,447	—	—	—	—	—	—	—	—	—
	180	5	0.0	1,358	344	0.0	1,819	248	0.0	2,754	246	—	—	—	5.4	*	631	2.7	10,147	551
		10	0.0	1,652	715	0.0	2,814	502	0.1	5,596	496	—	—	—	—	—	—	—	—	—
		25	0.0	3,143	1,873	0.0	8,402	1,274	0.1	14,631	1,254	—	—	—	—	—	—	—	—	—
		50	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
20	30	5	0.0	401	150	0.0	773	116	0.0	619	115	—	—	—	0.0	4,852	263	1.5	1,896	242
		10	0.0	473	316	0.0	876	235	0.0	917	232	—	—	—	32.9	*	612	8.5	5,107	492
		25	—	—	—	3.7	*	653	1.2	2,258	588	—	—	—	—	—	—	NA	*	1,273
		50	—	—	—	8.8	*	1,267	3.0	5,225	1,185	—	—	—	—	—	—	—	—	—
	90	5	—	—	—	0.0	2,367	253	0.1	2,029	245	—	—	—	13.0	*	635	4.4	*	543
		10	—	—	—	0.0	3,600	518	0.2	2,532	500	—	—	—	—	—	—	—	—	—
		25	—	—	—	111.1	*	1,383	9.4	8,806	1,270	—	—	—	—	—	—	—	—	—
		50	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
	180	5	—	—	—	0.0	6,237	448	0.2	5,709	436	—	—	—	—	—	—	—	—	—
		10	—	—	—	0.0	10,744	927	0.2	8,998	886	—	—	—	—	—	—	—	—	—
		25	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
		50	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
38	30	5	0.0	757	263	0.0	1,260	202	0.0	1,100	201	—	—	—	5.3	*	477	2.2	5,713	422
		10	0.0	907	559	0.0	2,118	410	0.1	2,420	405	—	—	—	804.6	*	1,050	34.3	10,246	858
		25	—	—	—	13.3	*	1,098	2.0	5,179	1,027	—	—	—	—	—	—	—	—	—
		50	—	—	—	182.3	*	2,163	7.4	*	2,069	—	—	—	—	—	—	—	—	—
	90	5	—	—	—	0.0	7,722	445	0.5	6,855	432	—	—	—	—	—	—	—	—	—
		10	—	—	—	0.0	10,533	914	0.6	10,289	881	—	—	—	—	—	—	—	—	—
		25	—	—	—	—	—	—	—	*	2,233	—	—	—	—	—	—	—	—	—
		50	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—

As compared to our previous results, the Benders decomposition structure enhances scalability. For instance, the Benders decomposition structure can now solve problems with 10 candidate lines, a 90-minute horizon and 50 scenarios, whereas the direct solution method ran out of memory. In large instances, however full column variable enumeration remains intractable.

In comparison, the column generation-based methods scale better and can solve larger instances to optimality. For instance, our exact algorithm terminates with the full set of 38 lines and a horizon of 30 to 90 minutes. The benefits are more significant when $K = 1$. In these instances, the vehicles can skip one reference stop, which translates into longer subpaths and hence exponentially more subpaths. In those instances, the combination of Benders decomposition and column generation can solve instances with up to 38 lines and a 30-minute horizon or 20 lines and a 90-minute horizon. These convergence improvements are driven by the fact that column generation requires 30 % fewer columns than the entire set to converge to the optimal solution.

Still, the scalability of column generation is hindered by the large number pricing problems per Benders sub-problem, which motivates our heuristic acceleration. In fact, our heuristic finds solutions for six more instances; in instances that are solved by both methods, the heuristic improves the solution by 4.5 % in average, while requiring 30 % less time and almost half of the Benders cuts. These benefits are, again, stronger when $K = 1$: as expected, the dominance criterion used in the heuristic label-setting algorithm becomes more impactful when the subpaths are longer. Note, moreover, that the heuristic does not significantly reduce the number of columns as compared to the full exact column generation algorithm. This suggests that the heuristic procedure can generate most of the relevant columns, thus enhancing scalability at limited costs in terms of solution quality.

In addition to the results reported in Table 5.4, we observed in our experiments that, for the methods using column generation, most of the time is spent in the pricing problem. Due to the large number of subproblems, an easy acceleration opportunity lies in computing parallelization, both across Benders sub-problems and across pricing problems. Viewed through this lens, the heuristic label-setting algorithm balances the time spent between (BMP) and (BSP), reducing the ratio of computational times from 85 to 24 on average. In addition, the Benders acceleration strategy that involves adding Benders cuts at the root node considerably speeds up the relaxation. Since the resulting gap is actually small, this strategy significnantly reduces the tree search.

5.5.3 Benefits of stochastic optimization

Finally, we estimate our stochastic optimization approach by reporting in Table 5.5 the Value of the Stochastic Solution (VSS) and the Expected Value of Perfect Information (EVPI) (Birge and Louveaux, 1997). The EVPI measures the cost of demand uncertainty as the difference between our solution and the Wait-and-See (WS) cost obtained by optimizing the reference lines in each scenario separately. The VSS measures the benefits of our stochastic optimization problem against the Expected Value (EV) solution that optimizes reference lines in a single representative scenario.

Table 5.5: Value of the stochastic solution, and expected value of perfect information.

				VSS			EVPI			
$\|\mathcal{L}\|$	Horizon	$\|\mathcal{S}\|$	z^*	EEV	VSS	$\|\frac{\text{VSS}}{z^*}\|$	WS	EVPI	$\|\frac{\text{EVPI}}{z^*}\|$	$\frac{\text{VSS}}{(\text{VSS}+\text{EVPI})}$
5	30	5	-3,034	-3,030	4	0.1%	-3,152	119	3.9%	3%
		10	-3,218	-3,125	93	2.9%	-3,338	120	3.7%	44%
		25	-3,580	-3,416	164	4.6%	-3,722	142	4.0%	54%
		50	-3,810	-3,464	346	9.1%	-3,970	161	4.2%	68%
	90	5	-9,859	-9,505	353	3.6%	-10,031	173	1.8%	67%
		10	-10,494	-9,916	579	5.5%	-10,674	180	1.7%	76%
		25	-11,051	-10,882	169	1.5%	-11,335	284	2.6%	37%
		50	-11,839	-11,460	379	3.2%	-12,129	290	2.4%	57%
	180	5	-18,492	-17,414	1,079	5.8%	-18,986	494	2.7%	69%
		10	-19,310	-18,722	587	3.0%	-19,858	548	2.8%	52%
		25	-19,941	-19,608	333	1.7%	-20,555	613	3.1%	35%
10	30	5	-5,405	-5,395	10	0.2%	-5,639	234	4.3%	4%
		10	-5,668	-5,380	288	5.1%	-5,899	232	4.1%	55%
		25	-6,329	-5,986	342	5.4%	-6,647	318	5.0%	52%
		50	-6,687	-6,419	268	4.0%	-7,062	375	5.6%	42%
	90	5	-17,181	-16,749	432	2.5%	-17,880	699	4.1%	38%
		10	-18,103	-17,240	863	4.8%	-18,908	804	4.4%	52%
	180	5	-32,153	-30,326	1,827	5.7%	-33,419	1,267	3.9%	59%
		10	-33,865	-31,694	2,171	6.4%	-35,181	1,316	3.9%	62%
20	30	5	-7,971	-7,475	496	6.2%	-8,191	220	2.8%	69%
20	30	10	-8,546	-8,180	367	4.3%	-8,776	229	2.7%	62%
20	90	5	-25,430	-24,351	1079	4.2%	-26,262	831	3.3%	56%
38	30	5	-10,671	-9,655	1016	9.5%	-10,726	56	0.5%	95%
38	30	10	-11,376	-10,326	1051	9.2%	-11,500	124	1.1%	89%

Note that the VSS can be significant—up to 9.5%. The expected value of perfect information ranges between 0.5% and 5.6%. In most cases, our stochastic optimization approach bridges most of the gap between the deterministic optimization baseline and the ideal case with perfect information, reflected through

the ratio $\frac{VSS}{(VSS+EVPI)}$ often exceeding 50%. These results highlight the benefits of our two-stage stochastic optimization formulation, reflecting the impact of demand uncertainty and variability in on-demand microtransit operations and its implications for system planning.

5.5.4 Practical assessment of demand-responsive microtransit

We compare our MiND-VRP solution to two practical benchmarks, reflecting fixed-route transit and ride-sharing. We define a fixed-route transit system as a single-stage variation of the MiND-VRP, where vehicles operate according to the reference schedule and second-stage deviations are not allowed. At the other extreme, ride-sharing is defined as a fully on-demand system without any reference schedule. We define ride-sharing benchmarks with vehicle capacities of 1, 2, and 4; we use the procedure from Bertsimas and Yan (2021) except that, instead of requiring all requests to be served, we maximize the number of served requests subject to fleet size constraints, and then minimize travel distance. To perform an apples-to-apples comparison, we consider out-of-sample data corresponding to five new weekdays; and we define microtransit, transit, and ride-sharing systems with the same total number of seats. For example, for a fleet of 10 microtransit vehicles each with a capacity of 10 passengers, the corresponding ride-sharing system will use 100 vehicles with a capacity of one, 50 vehicles with a capacity of two, and 25 vehicles with a capacity of four.

Figure 5.4 shows an overview of the performance of each system (demand coverage, level of service, and driving distance) as a function of capacity. Note, first, that microtransit achieves a high level of service overall, maintaining a waiting time under four minutes and a walking time under 30 seconds on average—corresponding to a walking distance of 40 meters. Level of service is further enhanced when the microtransit system exhibits higher flexibility ($K = 1$). On the negative side, microtransit services result in longer trip times due to the reliance on the reference line. Nonetheless, an average delay of 20 to 30 minutes for a transit service compared to a direct taxi is reasonable.

As expected, ride-sharing achieves a higher demand coverage with lower waiting times and lower delays. When the system operates with high capacity, ride-sharing benefits from the flexibility of matching passengers to nearby vehicles—as opposed to relying on pre-determined reference lines. On the other hand, when demand exceeds capacity, microtransit can result in fewer miles traveled. This is illustrated in Figure 5.5. In fact, microtransit can reduce the distance per passenger by over 50% as compared to single-occupancy ride-sharing, and by 25% as compared to higher-capacity ride-sharing. Another observation is that microtransit can actually reduce wait times as compared to high-capacity

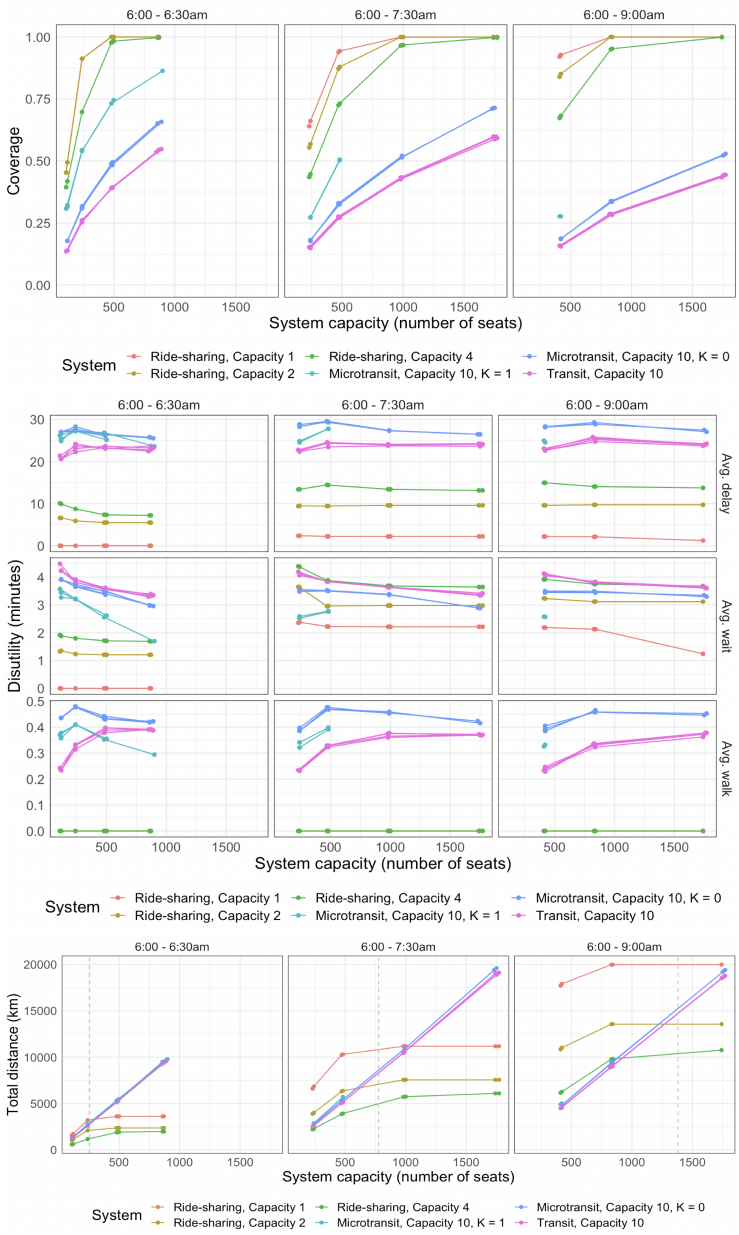


Figure 5.4: Comparison between fixed-route transit, microtransit, and ride-sharing. *Top:* Demand coverage. *Middle:* Average level of service. *Bottom:* Vehicle distance traveled. The dashed vertical line indicates the average number of requests in demand horizon.

ride-sharing, thus indicating potential benefits of consolidating demand around reference lines via the microtransit system. In other words, microtransit can achieve a significant reduction in vehicle miles traveled as compared to single-occupancy ride-sharing, while providing a comparable level of service to high-capacity ride-sharing.

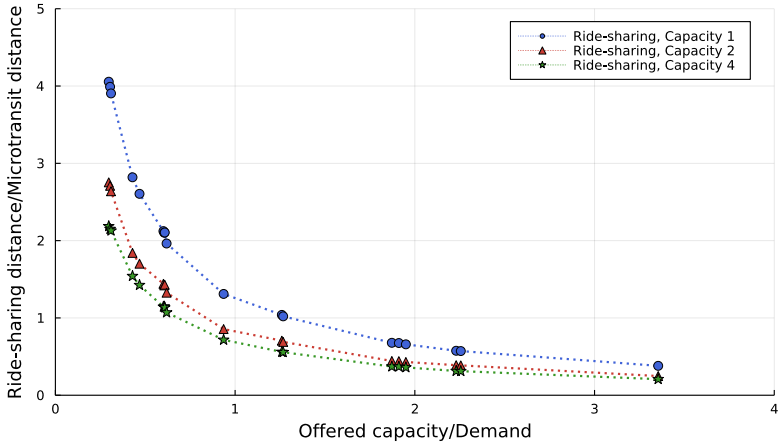


Figure 5.5: Comparison of distance traveled with ride-sharing versus microtransit, as a function of capacity-to-demand ratio.

As compared to fixed-route transit, microtransit increases demand coverage while maintaining short walking times and reducing wait times. The average arrival delay and walking time are both slightly higher under the microtransit system, because microtransit prioritizes serving more passengers (reflected in the hyperparameter M being larger than λ and μ), so the system covers requests that reduce the level of service on average. Nonetheless, these results underscore the benefits of microtransit systems in the public transit ecosystem. For example, with a total capacity of 1,500, the microtransit system can increase demand coverage by 25-30% over the full three-hour planning horizon, while maintaining a comparable level of service for the passengers that receive a service (higher delay by around 5 minutes, a few seconds of extra walk, lower wait time by 1 minute).

These benefits of microtransit stem from operational flexibility by means of on-demand deviations. The microtransit vehicles deviate from the reference line 3–7% of the time when $K = 0$; when $K = 1$, these deviations increase to 11% to 31% of segments. These numbers are, in fact, quite significant. Given the vehicle capacity of ten passengers, the number of deviations is bounded by ten segments, out of more than twenty segments per reference line in some cases—and multiple pickups can occur within each segment. The results indicate that

part of the demand can be covered without deviating, as in fixed-route transit, but that on-demand deviations are still leveraged to increase demand coverage and, to a smaller extent, enhance passenger service.

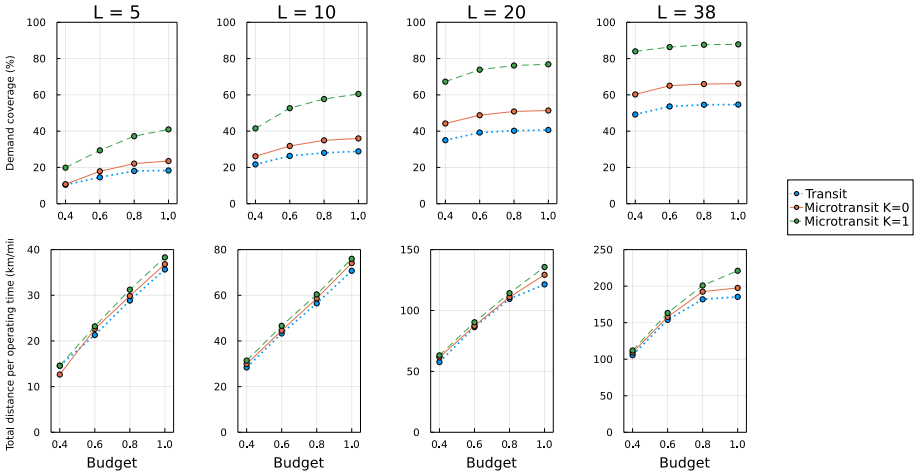


Figure 5.6: Comparison between fixed-route transit and microtransit systems.

Finally, Figure 5.6 reports demand coverage and driving distance with different budgets, measured as the proportion of the lines that can be operated. Note that microtransit serves 22% extra passengers even when vehicles need to visit each reference stop ($K = 0$), and 90% extra passengers when vehicles can skip one stop at a time ($K = 1$). These on-demand deviations barely increase vehicle miles traveled, especially with small line budgets. On average, microtransit distance is 3% higher when $K = 0$, and 8% when $K = 1$. Bringing these components together, note that a five-line microtransit system can serve a similar number of passengers as a twenty-line transit system, while reducing total distance by 68%. In other words, microtransit can provide Pareto improvements over existing transit solutions—higher demand coverage, lower operating costs, and less vehicle miles traveled with concomitant environmental benefits.

5.6 Conclusions

This paper studies the *Microtransit Network Design model for the Vehicle Routing Problem (MiND-VRP)*, which jointly optimizes network design and on-demand routing operations in hybrid microtransit systems featuring fixed-line components akin to public transit and flexible demand-responsive operations akin to ride-sharing. The model is formulated as a two-stage stochastic optimization problem with a bi-objective structure that maximizes demand coverage

and level of service, including walking and waiting time, and arrival delay. The model features a novel network representation of routing operations on a load-expanded network to maintain a tight second-stage formulation, by optimizing over subpath-based variables between reference stops as opposed to optimizing over arc-based variables. We have developed a solution algorithm that involves Benders decomposition to exploit the two-stage structure of the problem as well as column generation to generate subpaths iteratively. We also developed a tailored label-setting algorithm combined with a heuristic acceleration in order to generate subpath-based variables efficiently.

We applied the MiND-VRP using real-world data from New York City to model a shuttle service between Manhattan and LaGuardia airport. The system includes dozens of lines, hundreds of reference stops, and hundreds to thousands of passengers, resulting in up to 2.5 million first-stage variables and half a million pricing problems. Results demonstrate the scalability of the subpath-based formulation in the load-expanded network as well as the algorithmic approach combining Benders decomposition, column generation and our label-setting algorithm. From a practical standpoint, results suggest that microtransit can consolidate demand as compared to ride-sharing—resulting in fewer miles traveled and a competitive level of service under high demand—and achieve Pareto improvements as compared to fixed-route public transit—resulting in a higher demand coverage, lower operating costs, and environmental benefits.

This paper comes at a time when multiple hybrid solutions are emerging to enhance urban mobility, by combining the strengths of public transit and those of ride-sharing systems. This work can therefore be extended to reflect operating models where passengers have different origins and different destinations. From a technical standpoint, the modeling and algorithmic approach developed in this paper can be augmented with additional acceleration techniques in order to capture higher-dimensional problems—such as, for instance, larger line pools in the first stage. Finally, the potential benefits of microtransit systems outlined in this paper motivate the design of urban systems that would combine fixed-route transit options, hybrid microtransit systems, and ride-sharing vehicles. By proposing an integrated optimization framework, this paper provides methodological foundations and initial results to address this emerging class of problems.

References

Agarwal, S., Mani, D., and Telang, R. (2023). The impact of ride-hailing services on congestion: Evidence from indian cities. *Manufacturing & Service Operations Management*, (Articles in advance):1–22.

Ahuja, R. K., Magnanti, T. L., and Orlin, J. B. (1993). *Network flows : theory, algorithms, and applications*. Prentice-Hall.

Allen, D. J. (2017). *Lost in the transit desert: Race, transit access, and suburban form*. Routledge.

Alonso-Mora, J., Samaranayake, S., Wallar, A., Frazzoli, E., and Rus, D. (2017). On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment. *Proceedings of the National Academy of Sciences*, 114(3):462–467.

Alyasiry, A. M., Forbes, M., and Bulmer, M. (2019). An exact algorithm for the pickup and delivery problem with time windows and last-in-first-out loading. *Transportation Science*, 53(6):1695–1705.

Angelelli, E., Morandi, V., and Speranza, M. G. (2022). Optimization models for fair horizontal collaboration in demand-responsive transportation. *Transportation Research Part C: Emerging Technologies*, 140:103725.

Azadeh, S. S., van der Zee, J., and Wagenvoort, M. (2022). Choice-driven service network design for an integrated fixed line and demand responsive mobility system. *Transportation Research Part A: Policy and Practice*, 166:557–574.

Baaj, M. H. and Mahmassani, H. S. (1995). Hybrid route generation heuristic algorithm for the design of transit networks. *Transportation Research Part C: Emerging Technologies*, 3(1):31–50.

Baldacci, R., Mingozzi, A., and Roberti, R. (2011). New route relaxation and pricing strategies for the vehicle routing problem. *Operations research*, 59(5):1269–1283.

Balseiro, S. R., Brown, D. B., and Chen, C. (2021). Dynamic pricing of relocating resources in large networks. *Management Science*, 67(7):4075–4094.

Banerjee, S., Hssaine, C., Périvier, N., and Samaranayake, S. (2021). Real-time approximate routing for smart transit systems. *arXiv preprint arXiv:2103.06212*.

Barra, A., Carvalho, L., Teypaz, N., Cung, V.-D., and Balassiano, R. (2007). Solving the transit network design problem with constraint programming. In *11th World Conference in Transport Research-WCTR 2007*.

Barrena, E., Canca, D., Coelho, L. C., and Laporte, G. (2014). Single-line rail rapid transit timetabling under dynamic passenger demand. *Transportation Research Part B: Methodological*, 70:134–150.

Bertsimas, D., Jaillet, P., and Martin, S. (2019). Online vehicle routing: The edge of optimization in large-scale applications. *Operations Research*, 67(1):143–162.

Bertsimas, D., Ng, Y. S., and Yan, J. (2021). Data-driven transit network design at scale. *Operations Research*, 69(4):1118–1133.

Bertsimas, D. and Yan, J. (2021). The edge of optimization in large-scale vehicle routing for paratransit. *Preprint*.

Birge, J. R. and Louveaux, F. (1997). *Introduction to Stochastic Programming*. Springer-Verlag New York, Inc.

Borndörfer, R., Grötschel, M., and Pfetsch, M. E. (2007). A column-generation approach to line planning in public transport. *Transportation Science*, 41(1):123–132.

Borndörfer, R. and Karbstein, M. (2012). A direct connection approach to integrated line planning and passenger routing. In *12th Workshop on algorithmic approaches for transportation modelling, optimization, and systems*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

Boston Consulting Group (2019). On-demand transit can unlock urban mobility. https://www.bcg.com/publications/2019/on-demand-transit-can-unlock-urban-mobility.

Braverman, A., Dai, J. G., Liu, X., and Ying, L. (2019). Empty-car routing in ridesharing systems. *Operations Research*, 67(5):1437–1452.

Ceder, A. and Wilson, N. H. (1986). Bus network design. *Transportation Research Part B: Methodological*, 20(4):331–344.

Cipriani, E., Gori, S., and Petrelli, M. (2012). Transit network design: A procedure and an application to a large urban area. *Transportation Research Part C: Emerging Technologies*, 20(1):3–14.

Cummings, K., Vaze, V., Ergun, Ö., and Barnhart, C. (2023). Multimodal transportation alliance design with endogenous demand: Large-scale optimization for rapid gains. *arXiv preprint arXiv:2301.03414*.

Daganzo, C. F. and Ouyang, Y. (2019). A general model of demand-responsive transportation services: From taxi to ridesharing to dial-a-ride. *Transportation Research Part B: Methodological*, 126:213–224.

Desaulniers, G. and Hickman, M. D. (2007). Public transit. *Handbooks in operations research and management science*, 14:69–127.

Dunning, I., Huchette, J., and Lubin, M. (2017). JuMP: A modeling language for mathematical optimization. *SIAM review*, 59(2):295–320.

Eno Center for Transportation (2018). Uprouted: Exploring microtransit in the united states.

Galarza Montenegro, B. D., Sörensen, K., and Vansteenwegen, P. (2021). A large neighborhood search algorithm to optimize a demand-responsive feeder service. *Transportation Research Part C: Emerging Technologies*, 127:103102.

Galarza Montenegro, B. D., Sörensen, K., and Vansteenwegen, P. (2022). A column generation algorithm for the demand-responsive feeder service with mandatory and optional, clustered bus-stops. *Networks*.

Gehrke, S. R. and Reardon, T. (2018). Share of choices: Further evidence of the ride-hailing effect in metro boston and massachusetts. Technical report, Metropolitan Area Planning Council.

Hernandez, V. (2018). Metro's microtransit pilot program: Policy recommendations for equitable impact amongst low-income populations in los angeles. *Occidental College*.

INRIX (2022). Global traffic scorecard.

Karsten, C. V., Røpke, S., and Pisinger, D. (2018). Simultaneous optimization of container ship sailing speed and container routing with transit time restrictions. *Transportation Science*, 52(4):739–1034.

Ma, T.-Y., Rasulkhani, S., Chow, J. Y., and Klein, S. (2019). A dynamic ridesharing dispatch and idle vehicle repositioning strategy with integrated transit transfers. *Transportation Research Part E: Logistics and Transportation Review*, 128:417–442.

Magnanti, T. L. and Wong, R. T. (1984). Network design and transportation planning: Models and algorithms. *Transportation science*, 18(1):1–55.

Mangrum, D. and Molnar, A. (2017). The marginal congestion of a taxi in new york city. *Processed, Vamderbilt University*.

Marín, Á. G. and Jaramillo, P. (2009). Urban rapid transit network design: accelerated benders decomposition. *Annals of Operations Research*, 169(1):35–53.

McKinsey & Co. (2018). Travel and logistics: data drives the race for customers. https://www.mckinsey.com/industries/travel-logistics-and-infrastructure/our-insights/travel-and-logistics-data-drives-the-race-for-customers.

McKinsey & Co. (2021). Shared mobility: Where it stands, where it's headed. https://www.mckinsey.com/industries/automotive-and-assembly/our-insights/shared-mobility-where-it-stands-where-its-headed.

Mercier, A., Cordeau, J. F., and Soumis, F. (2005). A computational study of benders decomposition for the integrated aircraft routing and crew scheduling problem. *Computers and Operations Research*, 32(6):1451–1476.

Metropolitan Transportation Authority (2022). Static data feeds: New york city transit bus - manhattan. Acc. May 2022 at http://web.mta.info/developers/developer-data-terms.html#data.

Muter, I., Birbil, I., and Bülbül, K. (2018). Benders decomposition and column-and-row generation for solving large-scale linear programs with column-dependent-rows. *European Journal of Operational Research*, 264(1):29–45.

Muter, I., Birbil, S. I., and Bülbül, K. (2013). Simultaneous column-and-row generation for large-scale linear programs with column-dependent-rows. *Mathematical Programming*, 142(1-2):47–82.

Nourbakhsh, S. M. and Ouyang, Y. (2012). A structured flexible transit system for low demand areas. *Transportation Research Part B: Methodological*, 46(1):204–216.

NYC Taxi & Limousine Commission (2021). TLC Trip Record Data. Available at: https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page.

OECD (2017). Transition to shared mobility: How large cities can deliver inclusive transport services. In *International Transport Forum, Corporate Partnership Board Report, Paris, France, Tech. Rep.*

Ortega, F. A., Pozo, M. A., and Puerto, J. (2018). On-line timetable rescheduling in a transit line. *Transportation Science*, 52(5):1106–1121.

Özkan, E. and Ward, A. R. (2020). Dynamic matching for real-time ride sharing. *Stochastic Systems*, 10(1):29–70.

Papadakos, N. (2009). Integrated airline scheduling. *Computers and Operations Research*, 36(1):176–195.

Quadrifoglio, L., Dessouky, M. M., and Ordóñez, F. (2008). Mobility allowance shuttle transit (mast) services: Mip formulation and strengthening with logic constraints. *European Journal of Operational Research*, 185(2):481–494.

Quadrifoglio, L., Dessouky, M. M., and Palmer, K. (2007). An insertion heuristic for scheduling mobility allowance shuttle transit (mast) services. *Journal of Scheduling*, 10(1):25–40.

Quadrifoglio, L., Hall, R. W., and Dessouky, M. M. (2006). Performance and design of mobility allowance shuttle transit services: bounds on the maximum longitudinal velocity. *Transportation science*, 40(3):351–363.

Rahmaniani, R., Crainic, T. G., Gendreau, M., and Rei, W. (2018). Accelerating the benders decomposition method: Application to stochastic network design problems. *Siam Journal on Optimization*, 28(1):875–903.

Restrepo, M. I., Gendron, B., and Rousseau, L. M. (2018). Combining benders decomposition and column generation for multi-activity tour scheduling. *Computers and Operations Research*, 93:151–165.

Salazar, M., Rossi, F., Schiffer, M., Onder, C. H., and Pavone, M. (2018). On the interaction between autonomous mobility-on-demand and public transportation systems. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 2262–2269. IEEE.

Santi, P., Resta, G., Szell, M., Sobolevsky, S., Strogatz, S. H., and Ratti, C. (2014). Quantifying the benefits of vehicle pooling with shareability networks. *Proceedings of the National Academy of Sciences*, 111(37):13290–13294.

Shen, Y., Zhang, H., and Zhao, J. (2018). Integrating shared autonomous vehicle in public transportation system: A supply-side simulation of the first-mile service in singapore. *Transportation Research Part A: Policy and Practice*, 113:125–136.

Silva, D. F., Vinel, A., and Kirkici, B. (2022). On-demand public transit: A markovian continuous approximation model. *Transportation Science*, 56(3):704–724.

Steiner, K. and Irnich, S. (2020). Strategic planning for integrated mobility-on-demand and urban public bus networks. *Transportation Science*, 54(6):1616–1639.

Stiglic, M., Agatz, N., Savelsbergh, M., and Gradisar, M. (2018). Enhancing urban mobility: Integrating ride-sharing and public transit. *Computers & Operations Research*, 90:12–21.

Sun, L., Xie, W., and Witten, T. (2022). Distributionally robust fair transit resource allocation during a pandemic. *Transportation science*.

Szufel, P. (2022). Openstreetmapx julia package. Acc. Jan. 2022 at https://github.com/pszufe/OpenStreetMapX.jl.

The Economist (2018). Public transport is in decline in many wealthy cities. www.economist.com/international/2018/06/21/public-transport-is-in-decline-in-many-wealthy-cities.

Uber Movement (2020). New york city: Quarterly speed statistics by hour of day (q1 2020). Acc. Nov 2022 at https://movement.uber.com/cities/new_york/downloads/speeds?lang=en-US&tp[y]=2020&tp[q]=1.

US Department of Transportation (2016). Shared mobility current practices and guiding principles. Technical report.

US Environmental Protection Agency (2018). Sources of greenhouse gas emissions. https://www.epa.gov/ghgemissions/sources-greenhouse-gas-emissions.

Vazifeh, M. M., Santi, P., Resta, G., Strogatz, S. H., and Ratti, C. (2018). Addressing the minimum fleet problem in on-demand urban mobility. *Nature*, 557(7706):534–538.

Walteros, J. L., Medaglia, A. L., and Riaño, G. (2015). Hybrid algorithm for route design on bus rapid transit systems. *Transportation Science*, 49(1):66–84.

Wan, Q. K. and Lo, H. K. (2003). A mixed integer formulation for multiple-route transit network design. *Journal of Mathematical Modelling and Algorithms*, 2(4):299–308.

Wei, K., Vaze, V., and Jacquillat, A. (2022). Transit planning optimization under ride-hailing competition and traffic congestion. *Transportation Science*, 56(3):725–749.

Wellman, G. C. (2014). Transportation apartheid: the role of transportation policy in societal inequality. *Public Works Management & Policy*, 19(4):334–339.

Zeighami, V. and Soumis, F. (2019). Combining benders' decomposition and column generation for integrated crew pairing and personalized crew assignment problems. *Transportation Science*, 53(5):1479–1499.

Zhang, W., Jacquillat, A., Wang, K., and Wang, S. (2022). Routing optimization with vehicle-customer coordination. *Available at SSRN 4208397*.

Zhao, J. and Dessouky, M. (2008). Service capacity design problems for mobility allowance shuttle transit systems. *Transportation Research Part B: Methodological*, 42(2):135–146.

5.A Details on model formulation

5.A.1 Segment-based formulation

Throughout the section, we fix first-stage decisions $\boldsymbol{x}$ and $\boldsymbol{z}$, as well as scenario $s \in \mathcal{S}$.

We consider a road network over a set of nodes $\mathcal{N}$ that contains all stopping locations, i.e. all reference stops and passenger pickup locations. Each segment e in the set $\mathcal{E}_{\ell st}$ comprises a physical roadway $road(e)$, a travel time $tt(e)$, and a

Table 5.6: Additional inputs of the segment-based formulation.

	Type	**Description**
$\mathcal{N}$	Set	Nodes of roadway network: reference stops and pickup locations
$\mathcal{E}_{\ell st}$	Set	Time-load-augmented road segments corresponding to $(\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell$, $s \in \mathcal{S}$, where $e \in \mathcal{E}_{\ell st}$ is associated with a physical $road(e)$ and a travel time $tt(e)$
$\mathcal{T}$	Set	Set of time periods during the planning horizon
$\mathcal{P}_e$	Set	Passengers picked up on segment $e \in \mathcal{E}_{\ell st}$
$(\overline{\mathcal{V}}_{\ell st}, \bar{\mathcal{A}}_{\ell st})$	Graph	Time-load-expanded road network of trip $(\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell$ in scenario $s \in \mathcal{S}$
$\bar{\mathcal{A}}_e$	Set	Arcs in $\bar{\mathcal{A}}_{\ell st}$ corresponding to segment $e \in \mathcal{E}_{\ell st}$ for $(\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell$, $s \in \mathcal{S}$
$\bar{\mathcal{A}}^{idle}_{\ell st}$	Set	Arcs in $\bar{\mathcal{A}}_{\ell st}$ representing an idling vehicle, i.e. arcs that connect nodes corresponding to the same physical stop in $\mathcal{N}$ at consecutive time intervals
$\bar{\mathcal{A}}^{v}_{\ell st}$	Set	Arcs in $\bar{\mathcal{A}}_{\ell st}$ connecting the line's destination to the dummy sink node
$\mathcal{B}^{\kappa}_{\ell st}$	Set	Incoming arcs in the time-load-expanded network for reference trip $(\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell$ in scenario $s \in \mathcal{S}$, across reference stops κ through $\kappa+K$ for $\kappa \in \{2, \cdots, I_\ell - K\}$, at scheduled times $T_{\ell t}(\kappa) \cdots, T_{\ell t}(\kappa + K)$
τ^{walk}_{ep}	Parameter	Walk time of passenger $p \in \mathcal{P}_e$ via segment $e \in \mathcal{E}_{\ell st}$, $(\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell$, $s \in \mathcal{S}$
τ^{wait}_{ep}	Parameter	Wait time of passenger $p \in \mathcal{P}_e$ via segment $e \in \mathcal{E}_{\ell st}$, $(\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell$, $s \in \mathcal{S}$
$\overline{g}_a$	Parameter	Cost of arc $a \in \bar{\mathcal{A}}_{\ell st}$ on trip $(\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell$ in scenario $s \in \mathcal{S}$

set of passengers $\mathcal{P}_e$ who are picked up. The time horizon is discretized into T intervals in the set $\mathcal{T} = \{0, 1, \cdots, T\}$. The planning horizon spans the interval from the departure of the first trip from its origin ($t = 0$) to the arrival of the last trip at its destination ($t = T$).

To capture vehicle capacity constraints and time window constraints without relying on big-M constraints—thus retaining a tight second-stage formulation—we define a time-load-expanded network $(\overline{\mathcal{V}}_{\ell st}, \bar{\mathcal{A}}_{\ell st})$ for each reference trip $(\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell$ and each scenario $s \in \mathcal{S}$. The set $\overline{\mathcal{V}}_{\ell st}$ includes time-load-expanded nodes: each node $n \in \overline{\mathcal{V}}_{\ell st}$ is associated with a tuple (k_n, c_n, t_n), where $k_n \in \mathcal{N}^S$ denotes the corresponding location, $c_n \in \mathcal{C}$ tracks the passenger load, and $t_n \in \mathcal{T}$ is the the arrival time. The set $\overline{\mathcal{V}}_{\ell st}$ also includes a dummy sink node $\bar{v}_{\ell st}$ corresponding to the end of a trip, and source node $\bar{u}_{\ell st} = (\mathcal{I}^{(1)}_\ell, 0, T_{\ell t}(\mathcal{I}^{(1)}_\ell))$ corresponding to the beginning of the trip. We decompose the arc set $\bar{\mathcal{A}}_{\ell st} \subset \mathcal{V}_{\ell st} \times \mathcal{V}_{\ell st}$ into traveling arcs, idling arcs, and

terminating arcs, by writing $\bar{\mathcal{A}}_{\ell st} = \bigcup_{e \in \mathcal{E}_{\ell st}} \bar{\mathcal{A}}_e \cup \bar{\mathcal{A}}^{idle}_{\ell st} \cup \bar{\mathcal{A}}^{v}_{\ell st}$. Each traveling arc $a \in \bar{\mathcal{A}}_e$ corresponds to road segment $road(e)$ and tracks the incremental load $\sum_{p \in \mathcal{P}_e} D_{ps}$ and the incremental time $tt(e)$:

$$\bar{\mathcal{A}}_e = \left\{ (n,m) \in \overline{\mathcal{V}}_{lst} \times \overline{\mathcal{V}}_{lst} \,:\, (k_n, k_m) = road(e),\ c_m - c_n = \sum_{p \in \mathcal{P}_e} D_{ps},\ t_m - t_n = \lceil tt(e) \rceil \right\},\ \forall e \in \mathcal{E}_{\ell st}. \tag{5.41}$$

Next, each idling arc in $\bar{\mathcal{A}}^{idle}_{\ell st}$ connects nodes corresponding to two consecutive time intervals at the same physical stop:

$$\bar{\mathcal{A}}^{idle}_{\ell st} = \{(n,m) \in \overline{\mathcal{V}}_{\ell st} \times \overline{\mathcal{V}}_{\ell st} \,:\, k_n = k_m,\ c_n = c_m,\ t_m - t_n = 1\}. \tag{5.42}$$

Finally, each terminating arc in $\bar{\mathcal{A}}^{v}_{\ell st}$ connects the line's destination to the dummy sink node:

$$\bar{\mathcal{A}}^{v}_{\ell st} = \{(n,m) \in \mathcal{V}_{\ell st} \times \mathcal{V}_{\ell st} \,:\, k_n = \mathcal{I}^{(I_\ell)}_{\ell}, m = \bar{v}^{S}_{\ell st}\}. \tag{5.43}$$

Again, we can prune the time-load-expanded network by excluding intermediate nodes with no incoming or outgoing arcs, as well as the incident arcs. We define a segment-based cost $\overline{g}_a$ analogously to Equation (5.6) to capture passenger walking times, waiting times, and relative arrival delays:

$$\overline{g}_a = \begin{cases} \sum_{p \in \mathcal{P}_e} D_{ps} \left(\lambda \tau^{walk}_{ep} + \mu \tau^{wait}_{ep} + \frac{\tau^{delay}_{\ell tp}}{\tau^{direct}_{p}} \right) & \text{if } e \in \mathcal{E}_{\ell st}, a \in \bar{\mathcal{A}}_e \\ 0 & \text{if } a \in \bar{\mathcal{A}}^{idle}_{\ell st} \cup \bar{\mathcal{A}}^{v}_{\ell st}. \end{cases} \tag{5.44}$$

To ensure that the vehicle does not skip more than K reference stops in a row, we enforce that any subset of nodes corresponding to $K+1$ consecutive reference stops has at least one incoming arc. Let $\mathcal{B}^{\kappa}_{\ell st}$ be the set of incoming arcs in the time-load-expanded network across the reference stops k through $k+K$, at scheduled times $T_{\ell t}(k)$ through $T_{\ell t}(k+K)$.

$$\mathcal{B}^{\kappa}_{\ell st} = \bigcup_{\iota=\kappa}^{\kappa+K} \left\{ (n,m) \in \bar{\mathcal{A}}_{\ell st} \,:\, k_m = \mathcal{I}^{(\iota)}_{\ell}, t_m = T_{\ell t}(k_m) \right\}, \\ \forall (\ell,t) \in \mathcal{L} \times \mathcal{T}_\ell, s \in \mathcal{S}, \kappa \in \{2, \cdots, I_\ell - K\}. \tag{5.45}$$

We define the segment-selection decision variables:

$$\xi_a = \begin{cases} 1 & \text{if arc } a \text{ is selected, for } (\ell,t) \in \mathcal{L} \times \mathcal{T}_\ell, s \in \mathcal{S}, a \in \bar{\mathcal{A}}_{\ell st}, \\ 0 & \text{otherwise.} \end{cases} \tag{5.46}$$

The notation is summarized in Table 5.6. The second-stage segment-based formulation is given as follows for scenario $s \in \mathcal{S}$. Equations (5.47)–(5.49)

Table 5.7: Additional inputs of the path-based formulation.

Component	Type	Description
$\mathcal{Q}_{\ell st}$	Set	Valid paths for reference trip $(\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell$ and scenario $s \in \mathcal{S}$
$\mathcal{P}_q$	Set	Passenger pickup set corresponding to each path $q \in \mathcal{Q}_{\ell st}$
τ_{qp}^{walk}	Parameter	Walk time of passenger $p \in \mathcal{P}_r$ via path $q \in \mathcal{Q}_{\ell st}$, for $(\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell$, $s \in \mathcal{S}$
τ_{qp}^{wait}	Parameter	Wait time of passenger $p \in \mathcal{P}_r$ via path $q \in \mathcal{Q}_{\ell st}$, for $(\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell$, $s \in \mathcal{S}$
g_q^Q	Parameter	Cost of path $q \in \mathcal{Q}_{\ell st}$ on trip $(\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell$ in scenario $s \in \mathcal{S}$

are analogous to Equations (5.7), (5.10) and (5.11). Constraints (5.50) ensures that the vehicle does not skip more than K stops in a row.

$$\min \quad \sum_{(\ell,t)\in\mathcal{L}\times\mathcal{T}_\ell} \left(\sum_{a\in\bar{\mathcal{A}}_{\ell st}} \overline{g}_a \xi_a - M \sum_{p\in\mathcal{P}\,:\,(\ell,t)\in\mathcal{M}_p} D_{ps} \sum_{a\in\bar{\mathcal{A}}_e\,:\,p\in\mathcal{P}_e} \xi_a \right) \tag{5.47}$$

s.t.

$$\sum_{m:(n,m)\in\bar{\mathcal{A}}_{\ell st}} \xi_{(n,m)} - \sum_{m:(m,n)\in\bar{\mathcal{A}}_{\ell st}} \xi_{(m,n)} = \begin{cases} x_{lt} & \text{if } n = \bar{u}_{\ell st}, \\ -x_{lt} & \text{if } m = \bar{v}_{\ell st}, \\ 0 & \text{otherwise,} \end{cases} \tag{5.48}$$
$$\forall (\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell, i \in \overline{\mathcal{V}}_{\ell st}$$

$$\sum_{e\in\mathcal{E}_{\ell st}} \sum_{a\in\bar{\mathcal{A}}_e\,:\,p\in\mathcal{P}_e} \xi_a \le z_{plt} \qquad \forall p \in \mathcal{P}, (\ell, t) \in \mathcal{M}_p \tag{5.49}$$

$$\sum_{a\in\mathcal{B}^\kappa_{\ell st}} \xi_a \ge x_{lt} \qquad \forall \kappa \in \{2, \cdots, I_\ell - K\} \tag{5.50}$$

$$\xi_a \in \{0, 1\} \qquad \forall (\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell, a \in \bar{\mathcal{A}}_{\ell st} \tag{5.51}$$

5.A.2 Path-based formulation.

Throughout the section, we fix first-stage decisions $\boldsymbol{x}$ and $\boldsymbol{z}$, as well as scenario $s \in \mathcal{S}$.

Let $\mathcal{Q}_{\ell st}$ denote the set of all valid paths to reference trip $(\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell$ and each scenario $s \in \mathcal{S}$. Each path $q \in \mathcal{Q}_{\ell st}$ corresponds to a sequence of road segments that starts at the beginning of the line, end at its destination, satisfies flow balance in between, skips at most K reference stops in a row, does not pick up more than C passengers, and satisfies the time window constraints at the

reference stops. For each $q \in \mathcal{Q}_{\ell st}$, we store the passenger pickups in $\mathcal{P}_q \subset \mathcal{P}$. By definition, $\sum_{p \in \mathcal{P}_q} D_{ps} \leq C$. The cost g_q^Q of each path is:

$$g_q^Q = \sum_{p \in \mathcal{P}_q} D_{ps} \left(\lambda \tau_{qp}^{walk} + \mu \tau_{qp}^{wait} + \frac{\tau_{\ell tp}^{delay}}{\tau_p^{direct}} \right), \quad \forall (\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell, s \in \mathcal{S}, q \in \mathcal{Q}_{\ell st}. \tag{5.52}$$

We define the following decision variables:

$$\zeta_q = \begin{cases} 1 & \text{if path } q \text{ is selected, for } (\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell, s \in \mathcal{S}, q \in \mathcal{Q}_{\ell st}, \\ 0 & \text{otherwise.} \end{cases} \tag{5.53}$$

The notation is summarized in Table 5.7. The path-based formulation is given as follows. Equations (5.54) is analogous to Equation (5.7). Constraints (5.55) ensure that exactly one path is selected for each selected reference trip. Constraints (5.56) ensure that selected paths only serve passengers that have been assigned to that trip, analogously to Equation (5.11).

$$\min \quad \sum_{(\ell,t) \in \mathcal{L} \times \mathcal{T}_\ell} \left(\sum_{q \in \mathcal{Q}_{\ell st}} g_q^Q \zeta_q - M \sum_{p \in \mathcal{P} \,:\, (\ell,t) \in \mathcal{M}_p} D_{ps} \sum_{q \in \mathcal{Q}_{\ell st} \,:\, p \in \mathcal{P}_q} \zeta_q \right) \tag{5.54}$$

$$\text{s.t.} \quad \sum_{q \in \mathcal{Q}_{\ell st}} \zeta_q = x_{lt} \qquad \forall (\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell \tag{5.55}$$

$$\sum_{q \in \mathcal{Q}_{\ell st} \,:\, p \in \mathcal{P}_q} \zeta_q \leq z_{plt} \qquad \forall p \in \mathcal{P}, (\ell, t) \in \mathcal{M}_p \tag{5.56}$$

$$\zeta_q \in \{0, 1\} \qquad \forall (\ell, t) \in \mathcal{L} \times \mathcal{T}_\ell, q \in \mathcal{Q}_{\ell st} \tag{5.57}$$

5.B Preprocessing

5.B.1 Case study and algorithm parameters

The parameters used to define the case study and the algorithm characterization are shown in Table 5.8.

Exiting Manhattan for LaGuardia. First, we identified GPS locations for the set of four "exit points" from Manhattan toward LaGuardia Airport: the Queensboro Bridge, the Williamsburg Bridge, the Kennedy Bridge, and the Midtown Tunnel. Every reference line must leave Manhattan via one of these four exit points. From each, the driver heads directly to LaGuardia. We used Google Maps point estimates for travel times from each midpoint to LaGuardia during the morning rush to characterize the length of each terminating edge.

Travel times. We established all travel times across models and benchmarks using the `fastest_route` functionality provided by the OpenStreetMapX package in Julia (Szufel, P., 2022). To factor heavy Manhattan traffic into our travel time estimates, we designed a specially tuned speeds dictionary using speed data from Uber Movement (2020), rather than using the default speeds dictionary provided by OpenStreetMapX. We computed average speeds during the morning rush for each roadway type present in our Manhattan map (primary, secondary, tertiary, unclassified) and used these average speeds as input to the travel time estimation function.

Table 5.8: Parameter settings in case study.

Metric	Description	Value
Max walking distance	Computed between the passenger's origin and the station in an undirected version of the OSM graph	150 meters
Max waiting time	Computed as the difference between the arrival of the passenger to the stop and the pickup time	10 minutes
Max vehicle deviation	Defines the set of potential stops as the ones within X meters from the reference route	500 meters
Vehicle capacity (C)	Number of passengers that a vehicle can pick up	10
1^{st}-stage time discretization	Interval between first-stage time intervals. Defines the grouping of demand	3 minutes
2^{nd}-stage time discretization	Interval between second-stage time intervals. Used to do the routing adjustments (travel time between potential stops)	30 seconds
Walking time penalty (μ)	Penalty on one unit of walking time for each passenger	1
Waiting time penalty (λ)	Penalty on one unit of waiting time for each passenger	1
Failure to serve penalty (M)		100
Line budget (B_t)	Percentage of total line costs per time slot	60 %
Reference schedule buffer	Additional time added between reference stops, measured as +X% of the time at the nominal speed	20 %
Time interval duration	Time between time intervals in $\mathcal{T}_l$, defining the maximum frequency for a line	15 minutes

5.B.2 Reference line generation

We describe the process of generating the candidate line set $\mathcal{L}$. A high-quality set of candidate reference lines should span the service area. New York City public transit provides a natural set of transportation options satisfying this criterion. We began by obtaining the General Transit Specification Feed (GTFS) dataset for the Manhattan bus network from the MTA, i.e. the Metropolitan Transportation Authority (2022). The GTFS consists of several datasets describing the stops, trips, and timing of each bus line. We selected the subset of route shapes corresponding to trips departing during the weekday morning rush.

We aimed to avoid excessively long reference lines that duplicated the MTA network, and we wished to allow for more geographic diversity among the candidate

reference line set. To this end, we first broke each route shape in half for a total of 328 directed MTA route segments. After obtaining distances between each MTA route segment, we filtered connections that exceeded 1 kilometer. After building this directed graph, we conducted four breadth-first searches (BFS), each starting from one of the four LaGuardia exit points. Each leaf in the BFS tree corresponded to the starting segment of a candidate reference line, all terminating at the BFS root node. The final stop at LaGuardia was appended to each candidate line. This procedure resulted in 311 candidate lines.

After executing BFS, we post-processed the lines. We developed a new metric to measure the degree to which each reference line looped back in on itself, with the goal of automating the process of filtering out inefficient and meandering lines. Let d_{ij} be the distance between stops i and j and let δ be the average distance between consecutive stops in each reference line, with the n stops be ordered consecutively.

$$\frac{1}{n} \sum_{i+M<j} \mathbb{1}(d_{ij} \leq m\delta) \cdot (j-i)$$

Two stops that are supposed to be farther from each other—i.e., stop pairs with large values of $(j-i)$—incur higher penalties when they are geographically close to each other. The parameter M determines the radius of stop pairs to consider, and the factor m determines the level of "unacceptable" closeness, where $m \leq M$. We selected $m = 5$ and $M = 3$ for our purposes. After sorting the candidate reference lines on this metric, we selected the top 100 for further post-processing.

As a final post-processing step, we conducted a manual inspection of the 100 lines, discarding largely similar or impractical lines, to reach the final set of 38 candidate lines.

Part IV

Additional work

Chapter 6

A column-generation-based matheuristic for periodic and symmetric train timetabling with integrated passenger routing

Bernardo Martin-Iradi[a], and Stefan Ropke[a]

[a]DTU Management, Technical University of Denmark,
Akademivej Building 358, 2800 Kgs. Lyngby, Denmark

Status: Published in *European Journal of Operational Research.*

Abstract: In this study, the periodic train timetabling problem is formulated using a time-space graph formulation that exploits the properties of a symmetric timetable. Three solution methods are proposed and compared where solutions are built by what we define as a dive-and-cut-and-price procedure. An LP relaxed version of the problem with a subset of constraints is solved using column generation where each column corresponds to the train paths of a line. Violated constraints are added by separation and a heuristic process is applied to help to find integer solutions. The passenger travel time is computed based on a solution timetable and Benders' optimality cuts are generated allowing the method to integrate the routing of the passengers. We propose two large neighborhood search methods where the solution is iteratively destroyed and repaired into a new one and one random iterative method. The problem is tested on the morning rush hour period of the Regional and InterCity train network of Zealand, Denmark. The solution approaches show robust performance in a variety of scenarios, being able to find good quality solutions in terms of travel time and path length relatively fast. The inclusion of the proposed Benders' cuts provide stronger relaxations to the problem. In addition, the graph formulation covers different real-life constraints and has the potential to easily be extended to accommodate more constraints.

Keywords: Transportation, Periodic train timetabling, Matheuristics, Column generation, Passenger routing

6.1 Introduction

The planning process of railway companies is complex and is usually categorized into three main levels: *strategic*, *tactical* and *operational* (Bussieck et al., 1997). These levels form a hierarchical process used as a decision-making tool where each of the levels includes different problems whose solution is used as an input for the problems at the subsequent level as depicted in Figure 6.1.

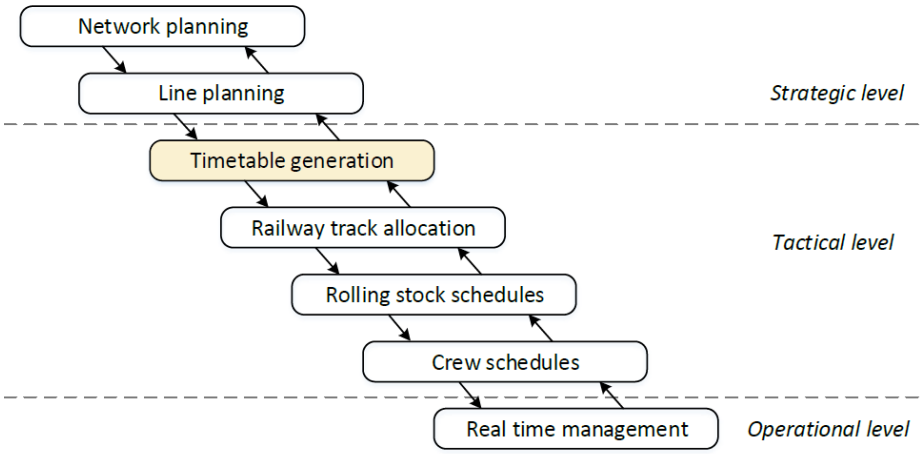


Figure 6.1: Railway planning process diagram adapted from Lusby et al. (2011)

In this study, the focus is mainly on the generation of timetables which is at the tactical level of the planning process. For that, the network and lines running on it, decided at the strategical level, are assumed fixed. The process of generating a timetable is formulated as the *Train Timetabling Problem* (TTP) and its main goal is to determine the arrival and departure times at the stations for each of the train lines.

The departure and arrival times are subjected to multiple track capacity constraints and specific requirements from the railway operating company. An obvious example of track capacity constraints is that two trains cannot be in the same track segment at the same time. In order to avoid having two trains at the same track segment at the same time, a *headway* is defined. The headway refers to the minimum time interval between two consecutive train movements and it is defined by the signaling system along the track. Likewise, a headway is defined for both departures and arrivals of consecutive trains along the same track segment. Moreover, a minimum dwell time is necessary to allow passengers to get on and off the train as well as changing drivers at specific stations. In the same way, minimum running times between two stations are limited by

the train speed, acceleration, breaking capabilities and an additional buffer time also known as *timetable margin*.

In general, the objectives are related to three main groups: customer satisfaction, robustness and cost-efficiency. These objectives may be conflicting in most cases. For instance, a timetable where all passengers have direct connections to their destinations at a high frequency would incur in an enormous operational cost for the train operating company (TOC). Therefore, a compromise between conflicting objectives should be found.

6.1.1 Focus of the paper

In this study, we focus on the generation of timetables from the passengers' point of view while also analyzing the robustness of the solution. The model presented relies in two main assumptions: (1) the running times between stations are considered fixed and (2) the timetable should be symmetrical or close to symmetrical (we elaborate on this in Section 6.3.3). The main contributions of the paper are two-fold: We present (1) a new graph formulation that allows us to directly generate non-conflicting schedules for all the trains of a line and also to include additional operational constraints with minor adaptations and, (2) a Benders' decomposition formulation that enables the integration of passenger routing in the timetabling generation process.

6.1.2 Paper structure

Section 6.2 lists several methods to solve the TTP through an extensive literature review. In Section 6.3 the model used and its characteristics are described. The solution methods used to solve the problem are described in Section 6.4, where each of the steps in the algorithms and how they interact together are carefully explained. Section 6.5 introduces the case studied, summarizes the computational results obtained from different tests and conducts an analysis of them. The paper concludes in Section 6.6 with a generic overview of the whole study and further study proposals.

6.2 Literature review

The literature about train scheduling is extensive. The different publications apply a wide range of methods to different cases. Some of them consider just a corridor or a junction whereas others study a whole network. Moreover, the nature of the resulting timetable (i.e., periodic or non-periodic) also affects the algorithm proposed. Several extensive surveys have been published (see Cordeau

et al. (1998), Caprara et al. (2007), Hansen (2009), Lusby et al. (2011), Cacchiani and Toth (2012) or Harrod (2012)).

Most of the studies that model a network assuming the periodicity of the timetable (periodic timetable) are based on the Periodic Event Scheduling Problem (PESP) first introduced by Serafini and Ukovich (1989). Odijk (1996) proposed a cutting plane algorithm to solve the PESP. Integer variables are used to ensure the travel intervals are respected and continuous variables to determine the arrival and departure times modulo the period. Later, Nachtigall (1998), Liebchen and Möhring (2002) and Peeters (2003) studied the Cycle Periodicity Formulation (CPF) that leads to a significant speed up in the solution times compared to earlier models. Given the effectiveness of the PESP, these models have been used to solve many network cases, whereas non-periodic approaches are used more often to model single-line corridors or congested networks where it may not be possible to schedule all trains in an efficient way.

Szpigel (1973) presented one of the first Integer Linear Program (ILP) formulations for the non-periodic TTP. The formulation is regarded as a job-shop scheduling problem where jobs (trains) need to be assigned to machines (track segments). Szpigel (1973) solved it using branch-and-bound applied to a Brazilian single-track line. Jovanovic and Harker (1991) proposed a Mixed Integer Linear program (MILP) formulation where the arrival/departure times are defined with continuous variables and the order of trains with binary variables and tries to find a reliable timetable. Carey and Lockwood (1995) proposed a mix of heuristic and branching procedure to solve a similar MILP as the one presented by Jovanovic and Harker (1991) in a one-way corridor, and Carey (1994) extended it to a two-way corridor showing that no additional constraints are needed. In general, most of the models proposed for solving non-periodic timetables are used for scheduling multiple competing timetables from different operators.

Furthermore, Brannlund et al. (1998) introduced a pure ILP formulation where the time was discretized and therefore, the formulation could be represented as a graph where the nodes represent the arrival and departure time instants to each station. This new formulation is referred to as *time-space graph* formulation but cannot be directly applied to large instances due to the large number of binary variables. As a result, further studying the LP relaxation of the model becomes more attractive and different methods have been developed based on it. The ILP formulation proposed by Caprara et al. (2002) defines a variable for each arc in the graph and it is solved using Lagrangian relaxation combined with sub-gradient optimization. Cacchiani et al. (2008) proposed a formulation where the variables refer to whole paths instead, and solved it applying column generation together with separation techniques. Cacchiani et al. (2010b) extended the formulation presented by Caprara et al. (2002) to be applied in

a network considering both passenger and freight trains and solved it using a similar procedure. Min et al. (2011) proposed a method for solving the train-conflict resolution problem with a column-generation based algorithm that takes advantage of the separability of the problem. Using a heuristic for the pricing problem (PP), the method is able to get near optimal conflict-free solutions in a few seconds. Cacchiani et al. (2013) applied dynamic programming to solve the clique constraints that arise in the graph formulations and developed an exact method whose performance is compared with various heuristics in Cacchiani et al. (2010a). Fischer (2015) formulates the TTP using a time-indexed graph and presents a method based on Lagrangian relaxation that improves the quality of the relaxation. Fischer and Schlechte (2017) extends the approach to also allow overtaking possibilities. Zhou et al. (2017) and Zhang et al. (2019) also take advantage of a graph formulation and effectively solve it using dual decomposition techniques. The methods are applied to the Beijing-Shanghai high speed corridor and show a better performance than the PESP model.

Last but not least, combining train timetabling and passenger routing has also been studied. Kinder (2008) extended the PESP model to a time-space graph and implemented an iterative approach where the timetable is re-planned after doing passenger routing. Gattermann et al. (2016) present an integrated model that finds timetables and passenger routes in which passengers are distributed temporally using *time-slices*. Borndörfer et al. (2017) also integrates timetabling and passenger routing in one model. The model tests and analyzes different passenger routing models on timetable optimization yielding significant improvements in travel time. Farina (2019) proposes a two-phase large neighborhood search heuristic for the combined train timetabling and passenger routing problem. The heuristic has similarities with the work presented in this paper, but employs different destroy and repair methods. Polinder et al. (2020) also implement a two-phase heuristic that aims at minimizing the passenger travel time. The method also accounts for the waiting time of the passengers at the stations and shows promising results in real-life instances. Several studies also refer to the problem at hand as the *demand-oriented train timetabling* problem. Li et al. (2017) implements a mixed integer quadratic model for the dynamic version of the problem and shows that it can effectively reduce the total passenger travel time. Zhou et al. (2019) studies passengers' booking decisions instead of the classic queue principle and uses a two-level method which combines a bi-level programming model with a priority-based heuristic which also shows benefits in terms of travel time for passengers.

6.2.1 Contribution and comparison to existing models

The modeling approach used in this paper is based on the time-space graph proposed in Caprara et al. (2002). As discussed in the literature review, this modeling approach has also been used in several later papers (e.g. Cacchiani

et al. (2008) and Cacchiani et al. (2010b)). In Caprara et al. (2002), the integer programming model was solved using a Lagrangian relaxation heuristic. The Lagrangian subproblem solves a longest path problem through an acyclic network. In Cacchiani et al. (2008), the problem was solved using column generation where the pricing problem also searches for longest paths through an acyclic network. We also solve the problem using column generation but use a pricing problem that can determine 1, 2 or 4 paths in one go. The pricing problem is solved as a standard shortest path problem (further details in Sections 6.3 and 6.4). This is possible due to tight frequency and symmetry constraints. There are several benefits of this approach: 1) The symmetry and frequency constraints are entirely handled in the pricing problem and fewer constraints are necessary in the master problem. 2) The LP relaxation produced by the master problem is potentially stronger compared to an approach that handles symmetry and frequency constraints in the master problem. 3) Fewer pricing problems must be solved. We believe that this is a major contribution of our paper.

Caprara et al. (2002) already constructed a cyclic timetable. We use this as a basis to generate cyclic timetables with a one hour period, useful for modeling the passenger train timetabling problem that a train operator faces. Normally, this problem is solved using a PESP model and, to the best of our knowledge, it is the first time that the time-space graph approach is used for this application.

The solution approach presented in this study constructs the timetable while considering the routing of the passengers. A routing sub-problem is used to generate Benders' cuts that guide the model to optimize the passenger travel time. As the literature review shows, this is an emerging topic in passenger train timetabling and we believe that the paper at hand proposes a simple but useful approach for integrating the passenger routing with the train timetabling problem.

The method proposed in the paper at hand is based on work done in the master's thesis of Bernardo Martin-Iradi (Martin-Iradi, 2018).

6.3 Problem formulation

The notation is based on the one from Cacchiani et al. (2010b). Let $S = \{1, ..., s\}$ denote the set of stations in the network. The network can be represented as a mixed multi-graph $N = (S, E \cup A)$ where each vertex $i \in S$ represents a station in the network and each edge $e = (h, i) \in E$ represents a single-track segment between two stations with no intermediate stations in between that is used by trains traveling in both directions (i.e. from h to i and from i to h). Finally, each arc $a = (h, i) \in A$ represents a double-track segment between stations h

and i with no intermediate stations that can be used only by trains traveling in one direction (i.e. from h to i). The graph can contain multiple arc/edges connecting the same two stations. For instance, in the network here studied there are segments with four tracks between two same stations (two in each direction). Therefore, the adjacent stations in between can be connected with four arcs (two in each direction) in the multi-graph. For convenience, for each station $i \in S$, let denote $\delta_N^+(i) \subseteq E \cup A$ the set of edges incident to i and arcs leaving i, and $\delta_N^-(i) \subseteq E \cup A$ the set of edges incident to i and arcs entering i. Furthermore, for both mono and bi-directional tracks, minimum time intervals between departures/arrivals (i.e., headway) on the same track are required. Therefore, for each $e \in E \cup A$ and station i of e, let denote:

- $d(i, e)$: minimum time interval between consecutive departures of trains traveling in the same direction from i on the track segment e.
- $a(i, e)$: minimum time interval between consecutive arrivals of trains traveling in the same direction at i on the track segment e.

Moreover, in the case of single-tracks, additional time interval requirements need to be set for trains traveling in opposite directions. Therefore, for each edge $e \in E$ and station i of e where $i \in \hat{S}$ and $\hat{S}$ is the set of stations connected by single-track segments, we denote:

- $f(i, e)$: minimum time interval between an arrival at i on e and a departure from i on e of trains traveling in opposite directions.
- $g(i, e)$: minimum time interval between a departure from i on e and an arrival to i on e of trains traveling in opposite directions.

Furthermore, let $S^* \subseteq \hat{S}$ be the stations only connected by single-track segments (i.e. a station that is adjacent to at least one single track segment and at least one double-track segment is placed in $\hat{S}$ but not in S^*, while a station that is adjacent to at least one single track segment and adjacent to zero double-track segments is placed in both $\hat{S}$ and S^*). Therefore, for station $i \in S^*$ we define:

- $h(i)$: minimum time interval between arrivals to i of trains arriving from any incident track segment.

In this case study, due to safety requirements, a minimum value of $d(i, e)$, $a(i, e)$ and $h(i)$ is defined, whereas $f(i, e) = 0$ and $g(i, e)$ is implicitly given by:

$$\text{g(i,e)} = \text{minimum travel time from i to h on e} + \text{minimum travel time from h to i on e},$$

where h is the other endpoint of e. The reason for $f(i, e) = 0$ is based on the rail infrastructure. At station i, each of the platforms has its own track and usually

the length of the tracks until their merging point allows a train to depart on e as soon as the other train has arrived from e.

6.3.1 Lines and timetables notation

The different lines link two major stations with a number of intermediate stations in between. Let $L = \{1, ..., l\}$ denote the number of operating lines in the network space and $D = \{1, 2\}$ the direction of the line, $D = 1$ for direction out of Copenhagen and $D = 2$ for direction towards Copenhagen. Let Υ be the set of trains that cover the L lines and D directions. For each train $j \in \Upsilon$, we denote l^j and d^j to its line and direction. Moreover, let f_j and e_j be the starting and ending station respectively and let $S^j := \{f_j, ..., e_j\} \subseteq S$ be the ordered set of stations visited by train j (stopping or not). Some segments between stations are formed by quadruple-track segments, meaning that each train can choose between two tracks to travel along that track segment. In this study, the quadruple-track segments connect various consecutive stations and it has been assumed that the train runs along the same track and cannot switch to the other track during the whole quadruple segment (see Figure 6.2). Let $N^j = (S^j, A^j)$ be the auxiliary network for each train $j \in \Upsilon$ where each arc in A^j is either an arc in A or an edge in E with an orientation, corresponding to the unique travel direction of j along the single-track. A timetable for each train is given by the departure time at f_j and the arrival time at e_j, and the arrival and departure times for the intermediate stations $S^j \backslash \{f_j, e_j\}$. Let $\phi_j(a)$ denote the

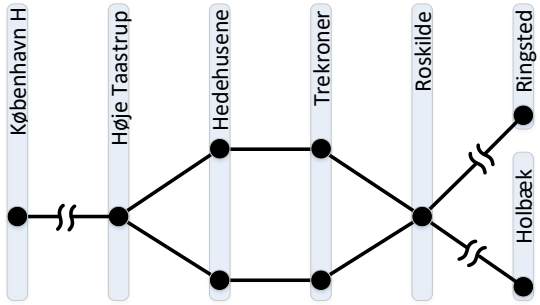


Figure 6.2: Illustration of the quadruple-track segment modelling in one direction.

running time along arc $a \in A^j$ of train $j \in \Upsilon$. Let $\omega_j^{min}(i)$ denote the minimum dwell time at station i for train $j \in \Upsilon$ where $i \in S^j \setminus \{f_j, e_j\}$. In the same way, there is an upper bound in the dwell time (i.e. $\omega_j^{max}(i)$) in the form of an additional percentage of the minimum dwell time ($\omega_j^{max}(i) \propto \omega_j^{min}(i)$). Note that, for a line containing N stations, there are N-1 minimum running times and N-2 minimum dwell times defined in one direction. The mentioned parameters

above are defined for each train meaning that the running and dwell time sets are defined independently for trains in different directions for the same line, as they may differ. Finally, the time horizon is defined as $T = \{1, ..., t\}$ referring to a whole hour discretized into time instants of half a minute ($|T| = 120$ time instants) and each line has an associated running frequency F^l indicating how many trains per hour cover each direction of that line.

6.3.2 A graph representation

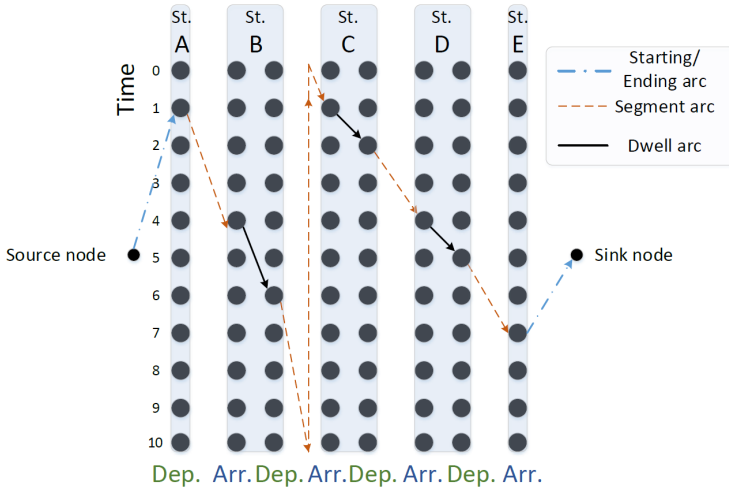


Figure 6.3: Graph representation of a train path with a time period of $|T| = 10$. The nodes at $t = 0$ correspond to a duplicate of the nodes of $t = |T|$.

The problem can be defined using graphs to represent the possible timetables (from now on referred to as *train paths*). Let $G = (V, R)$ be a directed and acyclic space-time graph. A sub-graph $G^j = (V^j, R^j)$ can be defined for each train $j \in \Upsilon$ (from now on referred to as *Train graph*) in which the nodes represent the arrivals or departures at a station at a given time instant. Figure 6.3 shows an example of a train path represented using a time-space graph.

The node set has the form

$$V^j = \{\sigma^j, \tau^j\} \cup \bigcup_{a=\{h,i\} \in A^j} (U_i^a \cup W_h^a)$$

where σ^j and τ^j are the *artificial source node* and *artificial sink node* respectively and the sets W_h^a for $h \in S^j \setminus \{e_j\}$ and U_i^a for $i \in S^j \setminus \{f_j\}$ represent the set of time instants where a train can depart from station h or arrive to station i on the track represented by arc $a \in A^j$ respectively (also called *departure* and *arrival* nodes). Let $u, w \in V^j$ be nodes of the node set

and let $\theta(u)$ be the time instant associated with node u. Furthermore, let $\Delta(u,w) := \theta(w) - \theta(u)$ denote the time interval between nodes u and w if $\theta(w) \geq \theta(u)$ and $\Delta(u,w) := \theta(w) - \theta(u) + T$ otherwise. Due to the periodic nature of the time horizon T, it is said that node u *precedes* or *coincides* with node w (i.e. $u \preceq w$) if $\Delta(w,u) \geq \Delta(u,w)$ as it is assumed that all the travel times used in this study case are far from the time horizon of one hour. Table 6.1 illustrates the time interval calculation with one example.

Table 6.1: Example of the time interval calculation between two nodes with a cycle time $|T| = 60$

$\theta(u)$	$\theta(w)$	$\Delta(u,w)$
10	15	5
15	10	55

For convenience, for each station $i \in S^j$, let denote $\delta^+_{N^j}(i) \subseteq A^j$ the set of arcs leaving i, and $\delta^-_{N^j}(i) \subseteq A^j$ the set of arcs entering i. The arc set R^j for each graph can be defined by four main types of arcs.

Starting arc set: These arcs connect the *artificial source node* with the set of nodes for the departure of the first station in the line. These arcs have a null cost (free arcs).

Segment arc set: These arcs connect the nodes related to the departure time from one station to the nodes related to arrival time to the next station in the line. Furthermore, the arc needs to satisfy that $\Delta(w,u) = \phi_j(a)$ where $\phi_j(a)$ denote the travel time for arc $a \in A^j$. The cost of the arc corresponds to the *travel time* between the departure and arrival instants in the respective sets.

Dwell arc set: These arcs connect the nodes related to the arrival time to one station with the nodes related to departure time from the same station in the line. Furthermore, the arc needs to satisfy that $\Delta(u,w) \in [\omega_j^{min}(i), ..., \omega_j^{max}(i)]$ for $i \in S^j \setminus \{f_j, e_j\}$. The cost of the arc corresponds to the *dwell time* between the arrival and departure instants in the respective sets.

Ending arc set: These arcs connect the set of nodes of the arrival to the last station in the line with the *artificial sink node*. These arcs have a null cost (free arcs).

As a result, the timetable for train $j \in \Upsilon$ is defined by any path from the artificial source node σ^j to the artificial sink node τ^j.

6.3.2.1 Main assumptions

The final graph formulation presented in this study is based on the assumption that the travel time of each train along each track segment joining two stations is fixed. In other words, it is not possible to slow down the train along the track segment and, therefore, the departure time from one station uniquely determines the arrival time at the next station. Even if slowing down is something that has to be done at the operational level, this assumption is supported by the fact that, in practice, slowing down a train between two stations in most cases is equivalent to forcing the train to stop in an endpoint station of the track segment for a longer time and then to travel at the regular speed along the track. This statement is not true in general but it holds for realistic cases. In particular, experimental results performed by Caprara et al. (2006) show that the solution values found by heuristic procedures are marginally affected by this additional constraint, whereas the corresponding running time per iteration is widely reduced, since the graph G turns out to be much smaller (for each train, the number of segment arcs between two stations is equal to the number of departure nodes). Furthermore, the above assumption simplifies the mathematical representation of the problem, yielding simpler and stronger overtaking and crossing constraints (see sections 6.3.4.3 and 6.3.4.4).

Another characteristic of the model assumed is the need for a symmetric timetable. When the train services are identical in both running directions it is easier to plan the timetable since the train path in one direction uniquely defines the path of the train in the opposite direction. Therefore, symmetric timetables are easier to plan and are more attractive to passengers as same transfer times are provided between pairs of trains in both directions (Liebchen, 2007). Nevertheless, this type of timetable reduces the degrees of freedom in the planning process and it is more suitable when the passenger demands are similar in both directions.

As a result, these two main assumptions can lead to a new, more efficient, graph formulation. On one side, keeping the running times fixed reduces the number of nodes to half since the arrival of a train is directly defined by the previous departure. On the other side, assuming symmetric paths for each line requires just creating one train path for a line, as the remaining line train paths are automatically defined.

6.3.3 Symmetric Line graph

The Symmetric Line graph formulation is defined, as the name states, per each line instead of per train, meaning that fewer graphs are needed. Ideally, each of the Symmetric Line graphs would include half of the nodes of one Train graph due to the fixed running times and symmetric paths. Nevertheless, in

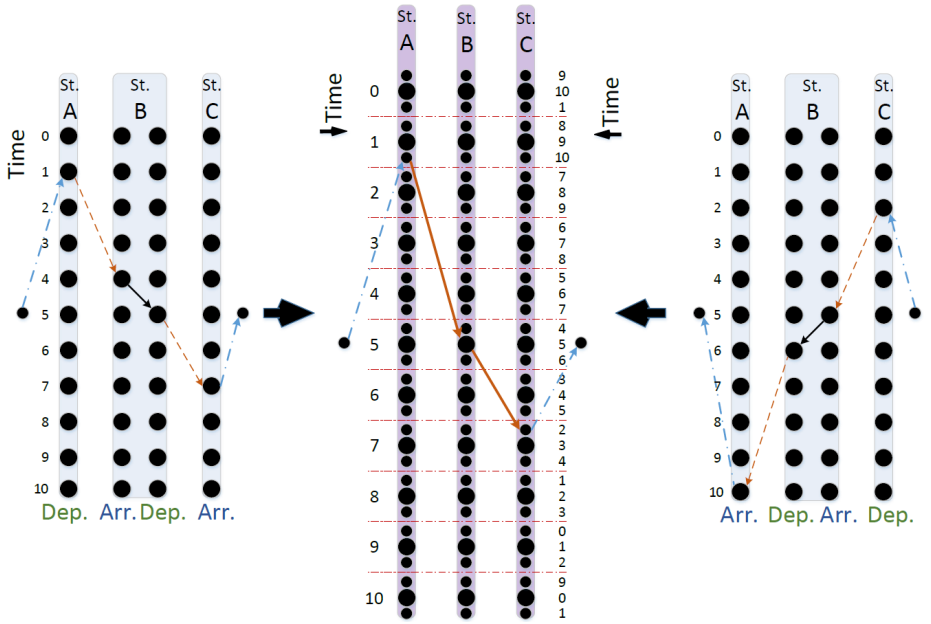


Figure 6.4: Representation of a path in the Symmetric Line graph as the combination of two paths in the respective Train graphs. In this example the symmetry gap is set to $\kappa = \pm 1$. The time axis on the left for the Symmetric Line graph denotes the departure time instants for the left train and the time axis on the right denotes the arrival times of the right train. The nodes corresponding to $t = 0$ are a duplicate of the nodes corresponding to $t = |T|$ and are added to help visualizing the symmetry of the paths in relation to the symmetry axis at $t = 5$.

practice, due to the nature of the infrastructure, the running times in opposite directions for a given track segment are sometimes slightly different, meaning that two exactly symmetrical paths cannot be achieved. Therefore, a *maximum symmetry gap* κ is considered. A line is considered symmetrical, if, for each station, the departure time of the train in one direction and the arrival time of the train in the opposite direction sum to the period time T. The symmetry gap adds flexibility to this and allows to also consider the line to be symmetrical if the sums of departure and arrival times are within the interval (i.e. $|T| \pm \kappa$). Figure 6.4 shows an example of two trains of a line that are considered symmetrical and their corresponding path in the new proposed graph. In this figure, the exactly symmetrical times at a station are depicted by larger nodes in the Symmetric Line graph and the symmetric instants that are within the gap considered (κ) are depicted with smaller nodes. The primary time axis indicates the departures times of the left-to-right train and the secondary one indicates the arrival times

of the right-to-left train. Starting with station A, the departure time of the left-to-right train is at time instant 1 and the arrival of the right-to-left train is at time instant 10. The sum of both times is 11 which is not equal to the planning horizon (10 in this case). Since the value is within the symmetry gap (10 ± 1) it is symbolized with a small node. For station B, the departure time of the left-to-right train is at time instant 5 and the arrival time of the right-to-left train is at time instant 5. The sum of both times is equal to 10 which is equal to the planning horizon meaning the departure and arrival of the trains are in perfect symmetry which is symbolized with the larger node. Last, in station C, the arrival of the left-to-right train is at time 7 whereas the departure of the right-to-left train is at 2. The sum of both times sums to 9, which is not perfectly symmetrical but again lies within the symmetry gap (10 ± 1) and therefore it is depicted as a small node.

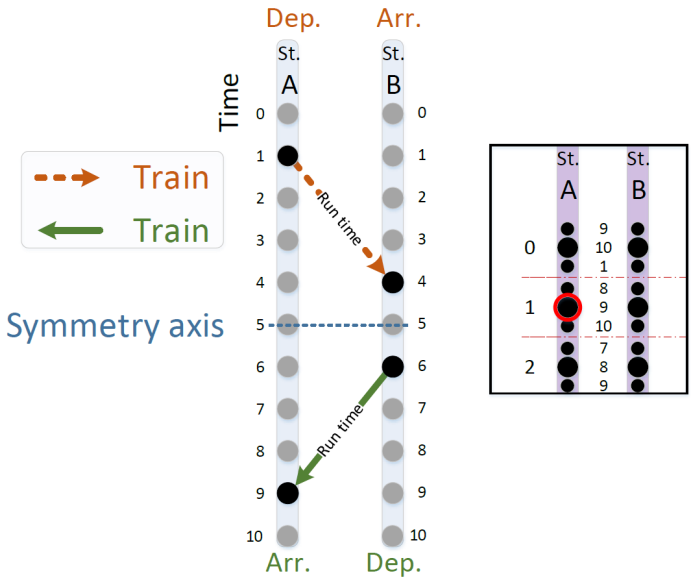


Figure 6.5: Representation of the Train graph nodes associated with one node (circled in red) of the Symmetric Line graph formulation. Notice that by assuming fixed running times the departure time from A directly defines the arrival time at B and vice versa. This example also shows that the train paths are perfectly symmetrical with respect to the symmetry axis. The nodes corresponding to $t = 0$ are a duplicate of the nodes corresponding to $t = |T|$ and are added to help visualizing the symmetry of the paths in relation to the symmetry axis at $t = 5$.

Each node in the graph represents the departure and arrival times of two sym-

metrical train paths of the same line along a track segment. In other words, one node from the Symmetric Line graph notation is equivalent to four nodes of the Train graph notation (see Figure 6.5). As we increase the symmetry gap, the amount of symmetrical departure and arrival time combinations increases in accordance. Since each of those combinations can be seen as a node in the graph, the growth is translated in $(1 + 2|\kappa|)$ nodes per time instant and station.

Regarding the arc set, the fact of assuming fixed running times allows us to merge the *segment* and *dwell* arc in a single *segment+dwell* arc. The weight of these arcs is given by the sum of running and dwell time for both trains. In order to avoid crossings or headway conflicts at a single-track segment, all arcs that result in incompatible departures, arrivals or crossings are not included in the graph. This ensures that all paths in the new graph correspond to feasible and compatible train paths for the line.

Regarding lines using the quadruple track segments (see Figure 2), it is assumed that trains make the same choice of track in both directions.

The output of the Symmetric Line graph corresponds to a set of compatible train paths covering the line. Depending on the nature and frequency of the line, the amount may vary between one, two or four train paths, as explained below:

If the line runs only during rush hour, trains only operate in one direction. This means that no symmetry is needed and a simple Train graph with fixed running times can be used. The output of it is just one train path, except if the frequency of the line is two trains per hour, then the output is two identical train paths exactly separated half an hour.

For regular lines, the output of the Symmetric Line Graph will be two symmetric train paths in opposite directions. If the frequency of the line is two trains per hour and direction, the output of the graph will correspond to two identical pairs of symmetric train paths separated by half an hour.

6.3.4 ILP formulation

In this section, the model is formulated as an ILP. In order to illustrate the different parts of the formulation, the notation of the Train graph is used. As it is explained in Section 6.3.3, the set of nodes of the Symmetric Line graph are formed by combinations of node sets from the Train graph formulation.

6.3.4.1 Formulation without track capacity constraints

The problem can be formulated as a version of the Set Packing Problem (SPP) that aims to minimize the sum of total path lengths. The binary variable $\lambda_q \in \{0,1\}$, $q \in Q$ defines if the group of line paths q is included in the optimal solution where Q is the set of possible line group paths. The parameter c_q denotes the cost of choosing the group of line paths $q \in Q$ that is the sum of path lengths. The formulation without the track capacity constraints is stated as follows:

$$min \quad \sum_{q \in Q} c_q \cdot \lambda_q \tag{6.1}$$

s.t.

$$\sum_{q \in Q^l} \lambda_q = 1 \quad \forall l \in L \tag{6.2}$$

$$\lambda_q \in \{0,1\} \quad \forall q \in Q \tag{6.3}$$

The objective function minimizes the cost (path lengths) of the solution train paths. Constraints (6.2) ensure that train paths are chosen to cover each line where Q^l is the set of possible line group paths for line $l \in L$ and constraints (6.3) state the binary property of the decision variable.

6.3.4.2 Headway constraints

Headway constraints are one of the track capacity constraints and ensure the minimum headway times between consecutive arrivals and departures at stations in the network.

$$\sum_{\substack{v \in U_i^a : v \preceq u \\ \Delta(v,u) < a(i,a)}} \sum_{q \in Q_v} \lambda_q \leq 1, i \in S, a \in \delta_N^-(i), u \in U_i^a, \tag{6.4}$$

$$\sum_{\substack{v \in W_i^a : v \preceq w \\ \Delta(v,w) < d(i,a)}} \sum_{q \in Q_v} \lambda_q \leq 1, i \in S, a \in \delta_N^+(i), w \in W_i^a, \tag{6.5}$$

$$\sum_{a \in \delta_N^-(i) \cap E} \sum_{\substack{v,u \in U_i^a : v \preceq u \\ \Delta(v,u) < h(i) \\ \theta(u) = t}} \sum_{q \in Q_v} \lambda_q \leq 1, i \in S^*, t \in T, \tag{6.6}$$

Let Q_v be the set of line group paths that use node v. Constraints (6.4) and (6.5) enforce that the minimum headway distance between consecutive arrivals and departures at each station respectively, of trains in the same direction, is respected. Moreover, constraints (6.6) ensure that in stations connected by single-track segments, the minimum headway between trains arriving to it is respected.

6.3.4.3 Overtaking constraints

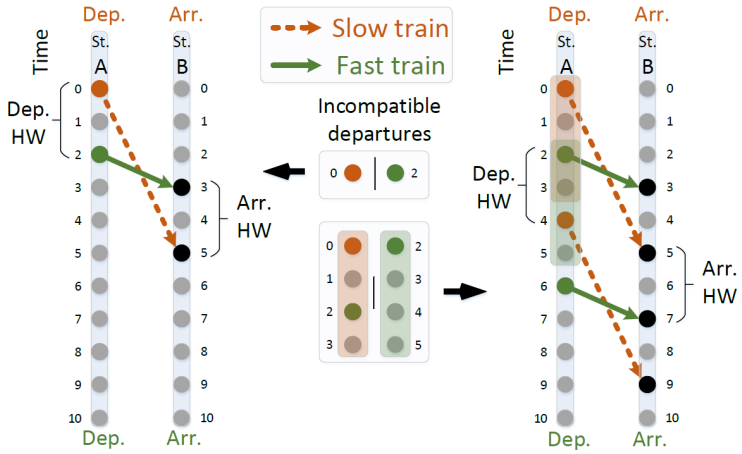


Figure 6.6: Illustration of an overtaking where $a(h,a) = 2$ and $d(i,a) = 2$. The left one is the simple version of the constraint while the right one is the stronger version implemented in this study.

It is not allowed that two trains traveling in the same direction on the same track overtake each other.

A basic example of an overtaking is shown on the left side of Figure 6.6 where both train departures are incompatible. The basic overtaking constraint would enforce that, at most, one slow train will depart from $t = 0$ or one fast train will depart from $t = 2$. In this study, a stronger version of this basic constraint is formulated based on the ones from Cacchiani et al. (2010b).

The following constraints (6.7) are defined for every pair of trains j, k along $a = (i,h)$ that is an arc in both auxiliary networks N^j and N^k. Moreover, j is considered the "slow" train and k is the train that can actually overtake it. Therefore, the travel time of train j should be greater than the one from train k (i.e. $\phi^j(a) > \phi^k(a)$). For a constraint, we define an earliest possible departure from i for trains j and k. These departure nodes are denoted v_1 and v_2 respectively. Node $v_1 \in W_i^a \cap V^j$ and node $v_2 \in W_i^a \cap V^k$ correspond to departure

nodes that are incompatible with each other (i.e. if train j departs at $\theta(v_1)$, then train k cannot depart at $\theta(v_2)$ and vice versa). The two trains j, k are considered incompatible when either $\min\{\Delta(v_1, v_2), \Delta(v_2, v_1)\} < d(i, a)$, meaning that their departures are too close in time or $\min\{\Delta(u_1, u_2), \Delta(u_2, u_1)\} < a(i, a)$ where u_1, u_2 are the respective arrival nodes for j, k corresponding to v_1, v_2, meaning that their arrivals to the next station are too close in time or $v_1 \prec v_2 \prec u_2 \prec u_1$ meaning that train k overtakes train j along the track.
Then, $v_3 \in W_i^a \cap V^j$ can be defined as the earliest possible departure of train j that is compatible with $\theta(v_2)$ such that $v_1 \prec v_3$. Analogously, $v_4 \in W_i^a \cap V^k$ can be defined as the earliest possible departure of train k that is compatible with $\theta(v_1)$ such that $v_2 \prec v_4$. It can be seen that any departure of train j from $[v_1, v_3)$ is incompatible with any departure of train k from $[v_2, v_4)$.

This stronger version of the constraint is illustrated in the right side of Figure 6.6. Let $Q_w^{l^j}$ be the set of line group paths that use node w and belong to train j of line l. Nodes v_1 and v_3 are depicted as the first and second slow train nodes in time respectively and nodes v_2 and v_4 are depicted as the first and second fast train nodes in time respectively. Note that in the illustration the minimum departure and arrival headway ($a(i, e)$ and $d(i, e)$) are respected for the trains but they overtake each other along the track.

$$\sum_{\substack{w \in W_i^a \cap V^j : \\ v_1 \preceq w \prec v_3}} \sum_{q \in Q_w^{l^j}} \lambda_q + \sum_{\substack{w \in W_i^a \cap V^k : \\ v_2 \preceq w \prec v_4}} \sum_{q \in Q_w^{l^k}} \lambda_q \leq 1, \forall j, k \in \Upsilon, v_1, v_2 \in W_i^a,$$

$$(\text{where } l^j \neq l^k, d^j = d^k,\ i, h \in S^j \cap S^k, a = (i, h) \in (A^j \cap A^k)) \tag{6.7}$$

6.3.4.4 Crossing constraints

It is not allowed that two trains traveling in opposite directions are on the same single-track segment at the same time.

A basic example of a crossing is shown on the left side of Figure 6.7 where both departures are incompatible. The basic constraint corresponding to this crossing would enforce that, at most, one slow or fast train will depart from $t = 0$. In this study, a stronger version of this basic constraint is formulated based on the ones from Cacchiani et al. (2010b).

The following constraints (6.8) are defined in a similar way to constraints (6.7). They are defined for every pair of trains j, k traveling in opposite directions such

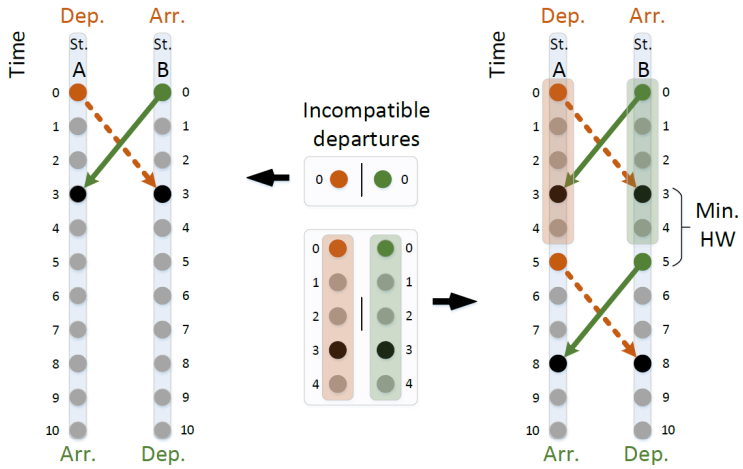


Figure 6.7: Illustration of a crossing where $f(h,e) = 2$. The left one is the simple version of the constraint while the right one is the stronger version implemented in this study.

that $a = (i,h)$ and (h,i) are arcs in the auxiliary networks N^j and N^k respectively and correspond to the set of edges E in the network. For a constraint, we define an earliest possible departure from i and h for trains j and k respectively. These departure nodes are denoted v_1 and v_2 respectively. Node $v_1 \in W_i^a \cap V^j$ and node $v_2 \in W_h^a \cap V^k$ correspond to departure nodes that are incompatible with each other (e.g. if train j departs at $\theta(v_1)$, then train k cannot depart at $\theta(v_2)$ and vice versa). The two trains j,k are considered incompatible when either $u_2 \preceq v_1$ and $\Delta(u_2,v_1) < f(i,e)$ or $u_1 \preceq v_2$ and $\Delta(u_1,v_2) < f(i,e)$, meaning that arrival to and departure from the same station are too close in time or $v_1 \prec u_2$ and $\prec v_2 \prec u_1$ meaning that train j and train k cross each other along the track.
Then, $v_3 \in W_i^a \cap V^j$ can be defined as the earliest possible departure of train j that is compatible with $\theta(v_2)$ such that $v_1 \prec v_3$. Analogously, $v_4 \in W_h^a \cap V^k$ can be defined as the earliest possible departure of train k that is compatible with $\theta(v_1)$ such that $v_2 \prec v_4$. It can be seen that any departure of train j from $[v_1, v_3)$ is incompatible with any departure of train k from $[v_2, v_4)$.

This stronger version of the constraint is illustrated in the right side of Figure 6.7. Nodes v_1 and v_3 are depicted as the first and second slow train nodes in time respectively and nodes v_2 and v_4 are depicted as the first and second fast train nodes in time respectively. Note that even if the minimum arrival headway $(f(h,e))$ is respected by the trains departing, they cross each other along the track.

$$\sum_{\substack{w \in W_i^a \cap V^j : \\ v_1 \preceq w \prec v_3}} \sum_{q \in Q_w^{l^j}} \lambda_q + \sum_{\substack{w \in W_h^a \cap V^k : \\ v_2 \preceq w \prec v_4}} \sum_{q \in Q_w^{l^k}} \lambda_q \leq 1, \forall j,k \in \Upsilon, v_1 \in W_i^a, v_2 \in W_h^a$$

$$(\text{where } l^j \neq l^k, d^j \neq d^k,\ i,h \in S^j \cap S^k, a = (i,h) \in A^j \cap E, (h,i) \in A^k \cap E) \tag{6.8}$$

6.3.4.5 Sibling constraints

There are specific pairs of lines that share identical or similar first and last stations but have slightly different stopping patterns. These pairs of lines (from now on referred to as *sibling lines*) should be spread along the cycle time as much as possible. In order to do so, the sibling constraints behave in the same way as the departure headway constraints (6.5). Let T_s denote the minimum time interval between consecutive departures of sibling lines in one direction at each station. Finally let $\Xi := \{(m_1, n_1), ..., (m_k, m_k)\}$ denote the set of sibling line pairs along the network where $m_k, n_k \in L$.

$$\sum_{\substack{v \in W_i^a : v \preceq w \\ \Delta(v,w) < T_s}} \sum_{q \in \{Q_v^{l^j} \cup Q_v^{l^k}\}} \lambda_q \leq 1, \quad \forall (l^j, l^k) \in \Xi, d \in D, w \in W_i^a,$$

$$(\text{where } j,k \in \Upsilon, i \in S^j \cap S^k, a \in \delta_N^+(i) \cap (A^j \cap A^k)) \tag{6.9}$$

Constraints (6.9) ensure that all the departures of sibling lines from any common station are spread at least a time interval of T_s in each direction.

6.3.5 Passenger routing model formulation

In order to route the passengers between stations, we introduce a multi-commodity flow problem (MCFP) formulation which is integrated with the ILP formulation by using a timetable solution as input information. Let $\bar{G} = (\bar{V}, \bar{A})$ be a graph formed by the set of nodes $\bar{V}$ and set of arcs $\bar{A}$. There is a node for each line $l \in L$, station $s \in S$ and time $t \in T$. We note that each node is used for both directions of a line. Let K be the set of commodities. We define each pair of *origin-destination* stations as a commodity $k \in K$ and the demand travelling between the corresponding origin and destination stations is given by an origin-destination (OD_k) matrix. Additionally, there is an artificial source and sink node o_k, d_k per commodity $k \in K$. The set $\bar{A}$ of passenger flow arcs is formed by different subsets:

- $\bar{A}_r \subseteq \bar{A}$: *Timetabling* arcs. Set of arcs corresponding to riding a timetabled train between consecutive stations. Due to the fixed running times, there are T arcs between consecutive stations per line and direction.
- $\bar{A}_d \subseteq \bar{A}$: *Dwell* arcs. Set of arcs corresponding to waiting time at a station, either dwelling on the train or waiting for the train to transfer to. There is one arc connecting two consecutive time instants in each station.
- $\bar{A}_s \subseteq \bar{A}$: *Source* and *sink* arcs. Set of arcs leaving one of the artificial source nodes o_k or entering one of the artificial sink nodes d_k. For any origin station i, the artificial source nodes of commodities having station i as *origin*, are connected with the departures of trains stopping at station i. Likewise, all the arrivals of trains stopping at station i are connected with the sink node of commodities that have station i as *destination*.
- $\bar{A}_t \subseteq \bar{A}$: *Transfer* arcs. Set of arcs to transfer between pairs of lines at a common station. For each node at a station with transfer options, there is one transfer arc to each train belonging to different lines, that also visit the station.

A small example of the different elements in the graph are shown in Figure 6.8. We identify a possible routing path for passengers travelling from station 1 and 4 (i.e. from o_k to s_k). This path consists of (1) boarding a train from line 1 that departs at time 1 from station 1, (2) getting off at station 3 and transferring to a train of line 2 that departs at time 4 and, (3) getting off at station 4 at time 5. Notice that, in this example, we consider a minimum transfer time of 2, meaning that the transfer arc will allow us to board a train at time 5 at earliest.

Let f_a^k be a variable that states if arc $a \in \bar{A}$ is used for commodity $k \in K$ and let t_a be the time to traverse arc $a \in \bar{A}$. To ease the problem formulation, we denote $\delta^+(v) \subseteq \bar{A}$ to the set of arcs leaving from node $v \in \bar{V}$ and $\delta^-(v) \subseteq \bar{A}$ to the set of arcs entering to node $v \in \bar{V}$. Finally, let x_a be a variable that defines if timetabling arc $a \in \bar{A}_r$ can be used. These variables refer to the timetable solution given and are kept constant in this problem. Notice that the formulation could be easily integrated with (6.1)-(6.9) by using a constraint that maps the λ_q variables to x_a. In order to avoid additional mathematical notation, this step has not been included.

The problem is formulated as follows:

$$\min \quad \sum_{k \in K} OD_k \sum_{a \in \bar{A}} t_a f_a^k \tag{6.10}$$

$$\sum_{a \in \delta^+(o_k)} f_a^k = 1 \quad \forall k \in K \tag{6.11}$$

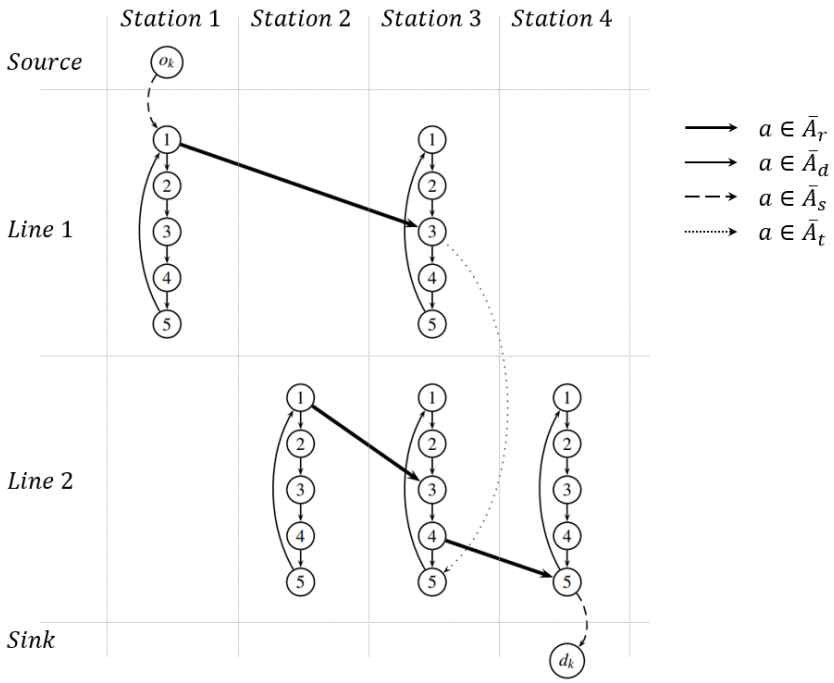


Figure 6.8: Passenger routing graph example with a cycle period of $|T| = 5$. In this case a subset of nodes and arcs is represented for trains of two different lines covering a subset of stations. The number in the node indicates a time instant. Only the artifical source and sink nodes for one commodity $k \in K$ are shown, which in this case, is the pair of stations $(1, 4)$.

$$\sum_{a \in \delta^-(d_k)} -f_a^k = -1 \quad \forall k \in K \tag{6.12}$$

$$\sum_{a \in \delta^+(v)} f_a^k - \sum_{a \in \delta^-(v)} f_a^k = 0 \quad \forall k \in K, v \in \bar{V} \backslash \{o_k, d_k\} \tag{6.13}$$

$$f_a^k - x_a \leq 0 \quad \forall k \in K, a \in \bar{A}_r \tag{6.14}$$

$$f_a^k \geq 0 \quad \forall k \in K, a \in \bar{A} \tag{6.15}$$

The objective function (6.10) minimizes the travel time of all passengers. Constraints (6.11) and (6.12) ensure that one arc is leaving from the source node and arriving to the sink node respectively for each commodity. Constraints (6.13) ensure the flow conservation and constraints (6.14) only allow to use arcs enabled by a timetable solution. Finally, constraints (6.15) define the variables

as linear positive. Notice that the capacity of the arcs is not limited, meaning that all passengers can board the same train. According to DSB (the train operator of the network studied), this is a fair assumption for this case. This method is based on studies such as the ones proposed by Schöbel and Scholl (2006) and Rezanova (2015). The problem can be decomposed into K different sub-problems, one per commodity and the totally unimodular structure of the problem formulation allows us to obtain an integer optimal solution by solving the LP model.

6.4 Solution method

Three solution methods are presented are based on what we call, a *dive-and-cut-and-price* procedure that heuristically solves the ILP formulation presented in Section 6.3.4. A Restricted Master Problem (RMP) is initialized with a subset of rows. Promising columns and violated cuts are added to it by column generation and separation procedure respectively in order to find an optimal LP solution. Then, branching is enforced through a dive heuristic in order to achieve integrality. Finally, the passengers are routed using the solution timetable and the travel time computed by solving (6.10)-(6.15).

Two of the methods are based on a large neighborhood search that iteratively transforms the solution by partially destroying and re-building it again. One of them uses the MCFP as a sub-problem to generate Benders' cuts for the RMP, helping to further integrate the passenger routing. The third method is a simple iterative process where a solution is fully constructed at every iteration.

Each of the steps in the methods is explained in detail in the following sections.

6.4.1 Column generation procedure

Taking into account the cycle time, the size of the network and the symmetry gap allowed, the number of possible line train paths to be considered is extremely large. In order to handle that amount of variables efficiently, column generation techniques are necessary.

A reduced version of the Master Problem (MP) is initially considered known as the Restricted Master Problem (RMP) that includes only a subset of the variables. These initial variables can just be a set of "dummy" artificial variables that satisfy the constraints of the RMP. For each line $l \in L$ a pricing problem is created (i.e. PP^l) that is in charge of providing line paths objects ($q \in Q^l$) that can potentially improve the current solution.

The formulation of the RMP is identical to the one of the original problem (see constraints (6.1)-(6.9)) except for the relaxed version of the decision variable (constraint (6.16)).

$$\lambda_q \geq 0 \quad \forall q \in Q \tag{6.16}$$

6.4.1.1 Pricing Problem

The goal of the PP is to find new promising train paths for the RMP. There is one PP per line and their function is to create a group of *line train paths* (referred to as a *column*) with the potential to improve the objective function. Here is where the Symmetric Line graph formulation described in section 6.3.3 becomes relevant. The use of a single graph for all the train paths of a line reduces the PP to a single *shortest path problem*. It can be noticed that the dual value of constraints (i.e. (6.4) - (6.9)) can be subtracted on the edge weights. Since, they are non-positive, we guarantee that the graph has always non-negative edge weights. Therefore, and knowing that the graph is directed acyclic (see Section 6.3.3), this problem can be solved using a dynamic programming algorithm.

Finally, to compute the reduced cost of a given path we need to subtract the dual value of constraint (6.2) for the given line, which is a real number and can lead to a final negative reduced cost. Every time the PP finds a column $q \in Q^l$ with a negative reduced cost, it is added as a new variable to the RMP and it is included in all the constraints where it has a non-zero coefficient.

6.4.2 Separation procedure

It is decided to add Constraints (6.7)-(6.9) by separation as the total amount is too large and only a reduced amount of them may be binding. The headway constraints are considered from the beginning in order to provide guidance to the column generation process.

Once the column generation procedure stops providing columns with negative reduced cost the separation procedure is applied. The separation of constraints (6.7)-(6.9) is done by enumeration and are checked in the same iteration. Every constraint that is violated by the current solution is added to the RMP.

Once the violated constraints are added to the model, the column generation procedure should be restarted. Adding more constraints to the model modifies the solution space and new columns with negative reduced cost can be found. The overall procedure of column generation and separation is summarized in Algorithm 6.1.

Algorithm 6.1 Column generation and Separation pseudo-code

```
1:  procedure colGenAndSep(fixedNodes)
2:      x = {}                                   ▷ start with empty solution
3:      PP ← fixedNodes                          ▷ fix nodes in graphs
4:      repeat
5:          repeat
6:              x ← solve(RMP)
7:              for all lines do
8:                  λ ← solve(PP(line))          ▷ generate a new column
9:                  if ĉ(λ) < 0 then
10:                     RMP ← λ                  ▷ add column with negative reduced cost
11:                 end if
12:             end for
13:         until no more columns with negative reduced cost
14:         RMP ← violatedConstraints(x)
15:     until no more violated constraints
16:     return x
17: end procedure
```

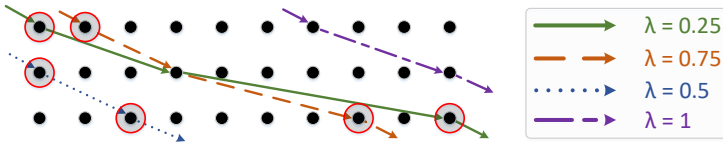


Figure 6.9: Fragment of a graph containing paths from a fractional solution where the nodes in the red circles are fractionally used.

6.4.3 Dive heuristic

The optimal solution for the MP can be fractional. In order to find an integer solution, a *dive* heuristic method is applied. The solution λ_q values are added to each of the graph nodes affected by that column. This measures the "usage" of each node and, if the solution is fractional, this means that some of the graph nodes are fractionally used (see Figure 6.9). The *dive* heuristic selects one of the fractionally used nodes and enforces to be part of the final solution, meaning that the final integer solution must contain that node. In order to do that, the shortest path problem is divided into two smaller and simpler ones where the chosen node works as the destination vertex in one of them and as the origin vertex in the other one. Apart from fixing the node, all the previously generated columns from the same graph that do not include the node need to be removed from the RMP. Once the heuristic step is concluded, the column generation should be started again as new promising columns may be

generated. One advantage of the dive heuristic is that it can lead faster to an integer feasible solution. A disadvantage of this method is that some branches of the tree are left unexplored and forcing the integrality of specific nodes that were fractional can lead to an infeasible final solution. This method only considers valid the solutions where all the trains of each line are scheduled. Therefore, in this study, if any column of the initial dummy set is part of a solution, the solution is considered infeasible. However, the initial set of dummy columns could potentially be used to allow solutions with fewer scheduled trains.

Most of the times, there are multiple fractionally used nodes in the solution and a criterion to select one is needed. In this study, we have opted for choosing any fractional node at random.

The procedure is summarized in Algorithm 6.2.

Algorithm 6.2 Dive heuristic pseudo-code

```
1: procedure DIVEHEURISTIC()
2:     [fixedNodes] = {}                          ▷ initialize empty list
3:     repeat
4:         x ← colGenAndSep(fixedNodes)           ▷ generate LP solution
5:         if x is fractional then
6:             [fixedNodes] ← newNode             ▷ fix a new node
7:         end if
8:     until x is integer or infeasible
9:     return x
10: end procedure
```

6.4.4 Passenger routing

The main objective of the model is to improve the passenger travel time (PTT). So far, the method minimizes the length of the train paths. This avoids extra additional dwelling of the trains at the stations and allows passengers traveling in the train to reach their destination fast. However, many passengers are required to transfer between trains to reach their destinations. Therefore, minimizing these transfer times becomes part of the overall objective of optimizing the passenger travel time.

Given a timetable solution to the RMP, the total passenger travel time is computed by solving (6.10)-(6.15).

6.4.5 Benders' cuts

After solving (6.10)-(6.15) for computing the total passenger travel time, we can generate a Benders' optimality cut for the RMP based on the dual values of the solution. A solution to the original RMP, fractional or not, always allows to route all the passengers and therefore, feasibility cuts are not generated. We define variable $z_k \geq 0$ for each commodity $k \in K$. Let $\pi_k^1, \pi_k^2 \in \mathbb{R}$ be the dual variables related to constraints (6.11) and (6.12) respectively. Additionally, we denote $\pi_{kv}^3 \in \mathbb{R}$ to the dual variable of constraint (6.13) and $\pi_{ka}^4 \leq 0$ to the dual variable of constraint (6.14). The arising optimality cut for each commodity k can be formulated as follows:

$$z_k \geq \pi_k^1 - \pi_k^2 + \sum_{a \in \bar{A}_r} \pi_{ka}^4 x_a \tag{6.17}$$

These cuts are added to the RMP and the objective function is updated to account for the z_k variables as follows:

$$\min \sum_{q \in Q} c_q \lambda_q + \alpha \sum_{k \in K} OD_k z_k \tag{6.18}$$

where $\alpha \in \mathbb{R}^+$ is a parameter that defines the weight of the passenger travel time in the objective function. The reader may wonder why we do not just minimize $\sum_{k \in K} OD_k z_k$ since this is the true objective considered in this paper (minimizing passenger travel time). The reason is that adding Benders' cuts slows the processing of each node in the dive-tree significantly, and therefore it is not possible to add all the Benders' cuts that actually are violated if we want the overall algorithm to finish within reasonable time (we use a one hour time limit in the computational results). Therefore, we only add a subset of the violated Benders cuts and the z_k variables are only an approximation of the true passenger travel time. This implies that it is beneficial to keep the path length component of the objective function as it guides the search towards solutions that also are attractive from a passenger travel time point of view. The cuts can be added by separation in the same way as the constraints mentioned in Section 6.4.2.

As a timetable solution is given as input, most of the arcs of the routing graph are not enabled, meaning that $x_a = 0$ in constraint (6.14). Looking at the objective function of the dual problem, which corresponds to the right-hand side of equation (6.17), the dual values of these arcs can fluctuate unrealistically as their value does not longer have an effect on the objective value of the dual problem. This can have a potential negative effect on the quality of the generated cuts. In order to avoid this, for all arcs where no flow should be allowed, we enable an infinitesimally small capacity ϵ. If x_a is not strictly zero for any arc in the graph anymore, the dual variables π_{ka}^4 are expected to be more realistic while still obtaining a near-optimal flow. This ϵ-capacity method is based on the *Kelley+* approach suggested by Fischetti et al. (2017).

6.4.6 Large neighborhood search

The main objective of the algorithm is to minimize the PTT. Therefore, every time a solution is computed, its PTT is compared with the best one found so far and updated if the new one is better. The process is framed in a Large Neighborhood Search (LNS) proposed by Shaw (1998) where the current solution is iteratively transformed into a different one. The transformation occurs by partially destroying the current solution and repairing it again. Our LNS is inspired by the work of Ropke and Pisinger (2006) and the whole process is summarized in Algorithm 6.3.

Algorithm 6.3 Large neighborhood search pseudo-code

```
1: procedure LNS()
2:     repeat
3:         x ← diveHeuristic()                    ▷ generate an initial solution
4:     until x is feasible
5:     x^b = x
6:     repeat
7:         x^t ← repair(destroy(x))               ▷ generate a new solution
8:         if x^t is feasible then
9:             c(x^t) ← routing(x^t)              ▷ compute PTT based on the routing of
   passengers
10:            if c(x^t) < c(x^b) then            ▷ compare passenger travel time
11:                x^b = x^t
12:                x = x^t                        ▷ only accept improving solutions
13:            end if
14:        end if
15:    until time limit
16:    return x^b
17: end procedure
```

The repair method is the already mentioned *dive-and-cut-and-price* whereas the the destroy method selects randomly ρ graph paths from the solution and removes them. This method is inspired by the ones implemented by Barrena et al. (2014). Furthermore, only solutions improving the PTT are accepted, adding relevance to the passengers' routes.

Two versions of the LNS method are implemented: (1) An LNS method without Benders' cuts and, (2) an LNS method with Beders cuts. In the latter, these cuts are added in the separation procedure, meaning that in line 14 of Algorithm 6.1, we compute equation (6.17) together with (6.7)-(6.9). The potential number of violated Benders' can be very large. This can result in not being able to solve the root node within the algorithm time limit. In order to avoid this, a internal

time limit is set to stop generating Benders' cuts.

Finally, in order to analyze the quality of the solution, this is compared with a lower bound solution. The lower bound (LB) value for the total path lengths is computed as the LP solution value at the root node in the initial dive heuristic (line 3 in Algorithm 6.3). The lower bound for the PTT is computed given a solution where all the trains operate at the minimum running and dwell times (i.e. shortest train paths) and passengers are able to transfer between any pair of lines at the minimum transfer time.

6.4.7 Random iterative method

An additional method to the LNS is proposed for comparison. The *dive-and-cut-and-price* procedure is repeated iteratively where each iteration is independent from the previous one. Since the randomness is introduced in the branching process, the root node is solved once and used as the re-start point at a new iteration. The entire method is summarized in Algorithm 6.4.

Algorithm 6.4 Random iterative method

```
1: procedure RandomIterative()
2:     x = {}                                   ▷ Initialize empty solution
3:     x^b = {}                                 ▷ Initialize best solution
4:     x^r ← solveRootNode(x)                   ▷ solve root node
5:     repeat
6:         x^t ← diveHeuristic(x^r)             ▷ apply dive heuristic
7:         if x^t is feasible then
8:             c(x^t) ← MCFP(x^t)               ▷ compute PTT based on the routing of
   passengers
9:             if c(x^t) < c(x^b) then          ▷ compare passenger travel time
10:                x^b = x^t
11:            end if
12:        end if
13:    until time limit
14:    return x^b
15: end procedure
```

6.5 Case study

The case studied here covers the Regional, Intercity and IntercityLyn (high-speed) network of Zealand, Denmark as seen in Figure 6.10. More specifically, the scope covers a one hour period during morning rush hour. This means that more lines run towards Copenhagen. Once, a timetable for this period is

obtained, it can be rolled out for the rest of the day by removing or adding rush hour lines.

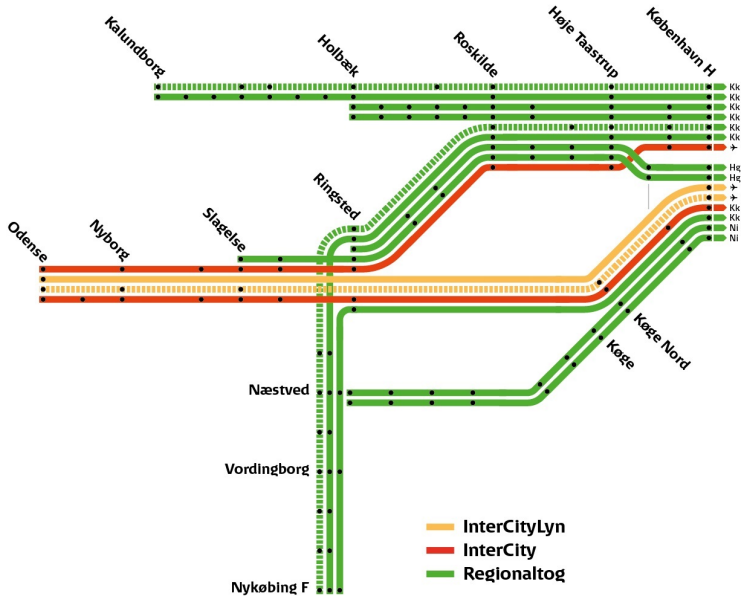


Figure 6.10: Network considered in the case study. Each line represents a frequency of one train per hour and direction and the dashed lines represent trains only running during rush hours (DSB, 2018)

The network is formed by 15 lines, covering 43 passenger stations. 3 of the lines only run during rush hour, which makes a total of 27 trains per hour to schedule. This translates in 12 Symmetric Line graphs, as two identical lines are handled as one line with a frequency of two trains per hour.

The number of tracks and the direction of trains running along them vary along each corridor. Three different types of track segments between stations are present in this network. A *single-track* segment, where trains can circulate in both directions but there can only be one train on the segment at a time. A *double-track* segment, where two tracks connect two stations allowing trains to travel in both directions (one track per direction) and a *quadruple-track* segment, formed by four tracks between two consecutive stations and trains can travel in both directions (two tracks per direction). The quadruple-track segments allow two trains going in the same direction to overtake each other along the segment.

In the network considered, there are two main single-track segments: the segment between Holbæk and Kalundborg and the segment connecting Køge Nord

and Næstved along the southern corridor. The rest of the network is connected by double-track segments with the exception of the segments between Høje Taastrup and Roskilde that are formed by quadruple-tracks.

The following input data has been provided by DSB, a danish TOC:

Minimum running time: This parameter states the minimum required time for a train to travel between two specific stations. This time interval is usually depending on the rolling stock type and the speed limits on the track segment. A value is given for every track segment connecting two consecutive stations in each line and direction.

Minimum dwelling time: This parameter states the minimum required time for a train to dwell at a specific station. This time is usually the time required by the passengers to board and leave the train. A value is given for every station visited by each line and each direction (i.e., between 30 seconds and 2 minutes).

Sibling lines: As mentioned in Section 6.3.4.5, there are specific pairs of lines that have similar or identical routes which are required by DSB to be as separated as possible in the timeline. There are three pairs of these lines considered in this case study. For example, the two lines reaching Kalundborg.

Minimum headway between trains: In this case study, three minimum headway values are given: 1) Minimum headway between two consecutive departing trains in the same track segment and direction, 2) minimum headway between two consecutive arriving trains in the same track and direction and 3) minimum headway between two consecutive trains arriving from single-tracks in opposite directions.

Single-platform stations: Some stations along the single-track segments have only one platform meaning that the station can only host one train at a time and a crossing between two trains is not allowed. It is assumed that, for the rest of stations in the network, any train arriving from an adjacent track segment has an available arriving platform.

Origin-Destination matrix: This matrix defines the number of passengers per hour traveling between each pair of stations. It does not consider passengers from stations outside the network (i.e., people entering the network from Germany or cities in Jutland). There is a total of 1806 pairs.

Station Clusters: A reduced version of the origin-destination pairs is proposed by defining a set of representative stations in the network, and clustering the neighboring ones. As shown in Figure 6.11, these stations correspond to the end-of line stations and stations where the track segments branch in different

directions (i.e., Roskilde and Køge Nord). The purpose of this network setup is only to reduce the number of commodities K considered when generating Bender's cuts while still capturing the routing decisions of most passengers. The evaluation of the total passenger travel time in the rest of the solution method is done using the entire network.

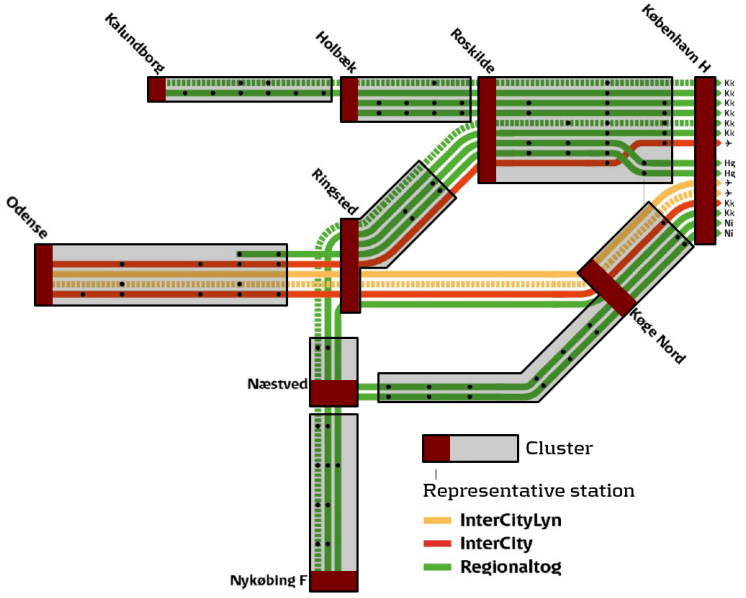


Figure 6.11: Network considered in the case study where the stations are divided into clusters, and the one marked in red is the representative station. Based on (DSB, 2018).

Minimum transfer time: In order for passengers to transfer between trains at a station, a minimum transfer time of 5 minutes is defined as a rule of thumb, meaning that if the time difference between the arrival of one train and the departure of another is lower, the transfer time corresponds to the time interval plus $|T|$.

The authors refer to Martin-Iradi (2018) for further details on the case study.

6.5.1 Instances

A number of instances are created based on the data from DSB. By changing the following four parameters, a total of 21 instances are obtained.

$\boldsymbol{HW_k}$: Minimum headway between consecutive arrivals and departures at Køben-

havn H. This station is seen as one of the most congested stations in the network where all lines stop at and, therefore, the headway at this station becomes interesting to analyze individually. This parameter measures in minutes the minimum interval between consecutive arrivals or departures at Københavns H in the same track segment.

$\boldsymbol{HW_n}$: Minimum headway between consecutive arrivals and departures at any station in the network. This parameter measures in minutes the minimum interval between consecutive train arrivals or departures at each track segment and station in the network.

$\boldsymbol{HW_s}$: Minimum headway between consecutive departures of sibling trains in the same direction from common stations. The pair of sibling lines may have slightly different stopping patterns or running and dwell times. This makes impossible to separate both train paths exactly half an hour during their entire trip. Therefore, a lower bound is needed that should be respected in any station. In this case, a minimum headway is defined for the consecutive departures from each station.

$\boldsymbol{\kappa}$: maximum symmetry gap in $\pm$ minutes between departure and arrival of trains in opposite direction belonging to the same line.

6.5.2 Computational results

The model has been entirely written in *Julia* language (Bezanson et al., 2017), modelled using *JuMP* (Lubin and Dunning, 2015) and using *CPLEX v. 12.9* as the solver. It has been tested in an Intel Xeon Processor X5550 (quad-core, 2.66 GHz) using one thread. Due to the large amount of parameter setting combinations, a base case is defined with the minimum values of each parameter (except for κ). Then, each parameter is tested independently keeping the others fixed. The parameter values for the base case are shown in Table 6.2. All instances are tested with a maximum dwell time of 3 minutes at each station. A parameter tuning has been conducted to determine the degree of destruction (ρ) of the destroy method which has been set to 5.

Table 6.2: Base case parameter setting

HW_k **(min)**	HW_n **(min)**	HW_s **(min)**	κ **($\pm$min)**
3	3	15	1.5

6.5.2.1 Impact of the pricing problem

In order to measure the benefits of the new graph formulation. A variant of the method (from now on referred as *Train-graph model*) is tested where the graphs only generate the train paths of a line in one direction and the symmetry is ensured by adding the respective constraints in the RMP. Due to the poor performance of the Train-graph model, only a comparison of the root node calculation is shown in Table 6.3. For the given network, the Symmetric Line graph is able to provide a stronger lower bound in significantly less time and fewer iterations. Actually, the lower bound of the Train-graph model corresponds to the sum of minimum running and dwell times of the trains to be scheduled (i.e. constant term).

Table 6.3: Root node results of the Symmetric Line graph model and the Train graph one.

Model	**Obj. value (min)**	**CG Iters**	**Time (s)**
Train graph	1981	39	27
Symmetric Line graph	1998.5	7	3

In addition, the dive heuristic based on the Train graph model is not able to find a feasible solution within the 1 hour limit. We believe that these results show that the graph formulation is an important part of the proposed solution method.

6.5.2.2 Instance results

The three solution methods presented are run 10 times for each scenario and the average values calculated. The time limit for each algorithm run is set to 1 hour and the internal time limit to stop adding Benders' cuts is set to 10 % of the algorithm time limit (i.e. 6 minutes). The value of α is set to the inverse of the number of passengers travelling within the time period.

Tables 6.4-6.7 show the results for each of the scenarios created by parameters HW_k, HW_n, HW_s and κ respectively. The first column indicates the solution method and the second one the parameter value of the scenario. The third and fourth columns display the best and average solution values of PTT respectively found across the 10 runs which are compared to the lower bound defined at the end of Section 6.4.6. The fifth and sixth columns indicate the average sum of path lengths (PL) relative to the best integer solutions with and without considering the fixed term compared to the lower bound defined at the end of Section 6.4.6. The seventh column displays the number of algorithm iterations

or equivalent repetitions of lines 7-14 in Algorithm 6.3 done per 1h run. The next three columns indicate the internal average iterations per algorithm iteration. First, the number of dive heuristic iterations which can be interpreted as the number of branches performed (i.e. nodes fixed). Next, the number of times the current LP solution is checked for violated constraints and, finally, the number of column generation iterations. The eleventh and twelfth columns shows the average number of columns and additional rows (6.7)-(6.9) added per algorithm iteration respectively. The thirteenth column indicates the number of Benders' cuts added in total. The next three columns indicate the proportional amount of time spent solving the RMP, PP and in the separation procedure respectively in relation to the total amount of time spent finding a solution. Last, the feasibility rate is stated that displays the proportion of algorithm iterations that result in a feasible integer solution. The average solution values are displayed in Figures 6.12-6.15 which also include results of a variant of the LNS method without Benders' cuts that only accepts solutions that improve the paths' length. More detailed results about this method variant can be found in Table 6.10 in Appendix 6.A.

The LNS-based methods that include passenger travel time in the acceptance criterion show a better performance in all instances. A main reason lies on the amount of iterations each method is able to perform. This suggests that partially destroying the solution is effective and enables exploring multiple neighborhoods. Table 6.8 shows the average solution quality, over all instances considered, in terms of passenger travel time and paths' length for the 4 variants of the solution method. The table shows that the addition of Benders' cuts results in a similar but marginally worse overall solution quality than the LNS method that did not use the Benders cuts. Adding these cuts increases the complexity of the RMP leading to fewer algorithm iterations and this causes the two methods to end with a similar solution quality.

All the methods find near optimal results both in PTT and path lengths in a reasonable amount of time for most of the scenarios. From Figures 6.12-6.15 a correlation between the length of the paths and the passenger travel time can be inferred. This is a realistic assumption since 95.8 % of all passengers in the network can reach their destination boarding a single train. However, optimizing the length of the train paths does not necessarily result in a shorter passenger travel time. This is deduced from the results of the LNS method with paths' length as acceptance criterion. In general, the solutions have indeed shorter train paths but the total passenger travel time is worse than for other methods. This shows that the simple integration of passenger travel time objective into the LNS method, through the acceptance criterion, is important in order to reach high quality solutions.

Intuitively, the parameter with the highest impact in PTT variation is the head-

way at the entire network, followed by the one at Copenhagen's central station. The similar performance for most of the values of HW_s indicates that this headway parameter has a very low impact in the solution space. The variability in the solution given by the randomness of the method allows, in cases like this where the instances are very similar, to have slightly better results even if the parameter value is more restrictive. Ideally this should not happen and we believe that a longer running time or more algorithm runs per instance would smooth the trend. Moreover, little variation in PTT is shown for the different values of maximum symmetry gap. This indicates a trade-off between the maximum gap allowed and the iterations the algorithm is able to perform within the time limit. A higher value of κ, expands the solution space but fewer algorithm iterations hinder the exploration of the neighborhood efficiently. On the other hand, if κ is too tight, the solution space becomes highly restricted and, regardless of the number of iterations, the solution quality decreases. It should be noted that the instances with the lowest values of the headway parameters correspond to the same instance. Different randomized seeds have been used in all cases and therefore, the results are not identical.

In terms of speed, it can be seen that the problem becomes harder to solve when increasing the parameter values. In particular, for high HW_n values, the LP becomes very hard to solve. Likewise, a higher value of κ, increases the complexity of the graph formulation and that is reflected in the time spent solving the pricing problems. Nevertheless, all methods are able to find solutions for $HW_k = 6$ minutes which is the maximum possible as 10 trains arrive per hour in København H through the same corridor. Also, solutions are found for values up to $HW_n = 5$ minutes and higher values were not further tested as they do not seem realistic for the network studied. Moreover, the algorithm finds solutions for $HW_s = 27$ minutes which seems to be the maximum allowed due to the differences in running times of the pairs of sibling lines. To put the solution values into perspective, we can compare them to the manual timetable planned by DSB ($PTT = 33.21, PL = 2049.5$). It can be noticed the presented methods produce better results for all instances. However, the manual timetable considers additional operational aspects such as rolling stock assignment and track crossings and therefore, we cannot see it as a fair comparison.

Table 6.4: Average performance of the algorithms for different values of HW_k.

Method	HW_k (min)	Best PTT gap (%)	PTT gap (%)	PL gap (%)	Var. PL gap (%)	Alg It.	Avg Dive It.	Avg Sep It.	Avg CG It.	Avg Columns	Avg +rows (%)	Benders' cuts	Avg RMP time (%)	Avg PP time (%)	Avg Sep time (%)	Feasibility rate (%)
LNS (with Benders' cuts)	3	2.54	2.75	0.35	40.0	80.6	2.4	6.0	125.0	299.5	1.41	124	38	34	13	91
	4	2.61	2.85	0.51	58.0	77.4	2.2	5.6	126.5	345.3	1.04	156	40	31	15	90
	5	2.72	2.94	0.86	98.0	83.3	2.0	5.0	107.6	315.8	0.79	182	39	32	14	90
	6	2.71	3.04	1.03	117.1	85.3	1.7	4.1	94.5	313.5	0.56	177	42	29	13	89
LNS (without Benders' cuts)	3	2.26	2.61	0.24	27.7	143.6	0.6	3.9	98.7	238.4	1.03	0	40	28	4	94
	4	2.37	2.83	0.55	62.9	134.1	0.7	3.8	115.2	294.7	1.08	0	43	30	4	88
	5	2.56	2.82	0.34	38.3	169.3	0.5	3.1	74.1	218.1	0.59	0	37	27	4	93
	6	2.71	3.00	0.79	90.0	164.2	0.6	2.9	79.2	245.8	0.44	0	40	27	3	92
Random Iterative	3	2.64	2.90	0.29	32.9	27.6	6.8	22.5	227.3	773.0	11.85	0	62	31	5	44
	4	2.82	3.11	0.54	62.0	19.0	7.0	23.2	272.3	999.1	9.82	0	72	24	3	45
	5	2.83	3.13	0.73	83.1	16.7	7.1	23.1	316.7	1251.6	8.97	0	85	13	1	41
	6	2.82	3.63	1.26	144.3	8.7	6.1	18.1	351.7	1688.6	4.93	0	87	12	1	36

Table 6.5: Average performance of the algorithms for different values of HW_n.

Method	HW_n (min)	Best PTT gap (%)	PTT gap (%)	PL gap (%)	Var. PL gap (%)	Alg It.	Avg Dive It.	Avg Sep It.	Avg CG It.	Avg Columns	Avg +rows (%)	Benders' cuts	Avg RMP time (%)	Avg PP time (%)	Avg Sep time (%)	Feasibility rate (%)
LNS (with Benders' cuts)	3	2.54	2.78	0.40	45.7	74.9	2.2	5.9	125.5	304.5	1.44	119	38	35	14	90
	3.5	2.51	2.86	0.69	78.6	72.3	2.1	5.6	120.8	320.0	1.29	118	39	34	14	92
	4	2.69	2.99	0.94	107.7	55.8	2.2	6.1	147.2	464.5	1.85	156	46	31	14	88
	4.5	2.96	3.27	1.32	151.1	48.5	2.3	7.5	220.3	851.8	3.51	131	66	21	11	76
	5	2.84	3.59	1.06	121.4	25.3	3.1	10.6	393.3	1620.5	7.13	137	84	7	8	78
LNS (without Benders' cuts)	3	2.36	2.61	0.28	32.0	139.4	0.6	3.8	100.7	242.4	1.04	0	41	28	4	95
	3.5	2.43	2.68	0.47	53.1	128.5	0.5	3.5	104.5	276.5	1.03	0	52	25	3	93
	4	2.46	2.89	0.89	101.1	123.3	0.5	3.2	107.7	311.3	1.01	0	45	30	4	91
	4.5	2.76	3.12	0.75	85.7	96.7	1.0	5.7	149.7	513.5	3.73	0	70	22	2	83
	5	2.82	3.43	1.09	124.0	66.9	1.7	9.6	323.4	1273.7	6.91	0	93	6	0	88
Random Iterative	3	2.64	2.90	0.29	32.9	28.0	6.9	22.6	227.9	776.3	11.86	0	61	32	5	43
	3.5	2.73	3.19	0.78	89.4	15.9	6.5	22.6	290.4	1084.8	14.10	0	87	11	1	41
	4	2.83	3.44	1.06	120.6	7.5	6.1	23.9	388.8	1576.7	16.64	0	92	7	1	41
	4.5	3.13	3.54	1.44	164.3	5.1	5.4	22.1	484.0	1975.8	18.91	0	94	5	0	35
	5	3.47	3.78	1.21	138.3	3.8	4.8	18.8	492.7	2247.1	17.78	0	97	3	0	35

Table 6.6: Average performance of the algorithms for different values of HW_s.

Method	HW_s (min)	Best PTT gap (%)	PTT gap (%)	PL gap (%)	Var. PL gap (%)	Alg It.	Avg Dive It.	Avg Sep It.	Avg CG It.	Avg Columns	Avg +rows (%)	Benders' cuts	Avg RMP time (%)	Avg PP time (%)	Avg Sep time (%)	Feasibility rate (%)
LNS (with Benders' cuts)	15	2.60	2.81	0.47	54.0	81.9	2.2	5.8	130.0	312.2	1.38	129	40	32	13	92
	17	2.51	2.68	0.44	49.7	75.6	2.4	6.1	133.9	329.3	1.62	142	39	33	14	89
	19	2.39	2.72	0.33	38.0	84.0	2.4	5.8	119.7	298.2	1.64	123	37	33	14	92
	21	2.37	2.80	0.37	42.0	75.2	2.1	5.5	131.2	352.8	1.66	135	42	31	13	91
	23	2.50	2.69	0.29	33.4	81.3	2.4	5.7	113.2	295.1	1.90	150	37	32	15	91
	25	2.77	3.00	0.51	58.3	86.5	1.9	5.5	113.2	281.1	2.12	115	36	34	15	89
	27	2.53	2.98	0.45	51.7	69.9	1.9	6.0	131.6	361.4	3.30	112	37	36	16	78
LNS (without Benders' cuts)	15	2.26	2.61	0.24	27.7	142.2	0.6	3.9	98.7	238.3	1.03	0	41	28	4	94
	17	2.45	2.65	0.28	32.3	139.3	0.7	3.9	102.6	245.4	1.22	0	41	29	5	92
	19	2.49	2.72	0.37	42.6	136.9	0.6	3.7	104.9	246.2	1.27	0	41	29	5	90
	21	2.49	2.89	0.39	44.3	126.1	0.8	4.5	110.5	278.3	2.00	0	41	32	6	88
	23	2.37	2.68	0.21	24.3	155.3	0.4	3.3	85.2	210.5	1.38	0	37	27	7	92
	25	2.62	2.88	0.44	50.3	132.3	0.5	3.9	106.8	251.3	1.99	0	39	29	8	91
	27	2.71	2.96	0.57	65.1	133.3	0.5	3.6	94.8	236.2	2.29	0	38	29	9	88
Random Iterative	15	2.64	2.87	0.39	44.6	28.0	6.9	22.5	222.6	763.0	11.92	0	61	31	5	43
	17	2.86	3.04	0.55	62.5	24.3	7.0	23.4	229.5	835.1	13.01	0	84	13	2	53
	19	2.86	3.15	0.47	53.2	24.6	6.7	21.4	215.9	777.6	12.76	0	79	18	3	31
	21	2.94	3.21	0.50	56.8	19.3	7.1	23.6	288.3	989.1	14.52	0	87	11	1	35
	23	2.75	3.01	0.48	54.3	23.1	6.1	19.8	202.9	798.0	13.80	0	84	13	2	49
	25	2.95	3.22	0.67	76.4	16.3	6.3	20.7	278.7	1090.8	15.49	0	91	8	1	48
	27	2.79	3.19	0.57	65.0	36.3	4.2	13.2	143.3	662.3	14.10	0	80	16	3	13

Table 6.7: Average performance of the algorithms for different values of κ.

Method	κ (± min)	Best PTT gap (%)	PTT gap (%)	PL gap (%)	Var. PL gap (%)	Alg It.	Avg Dive It.	Avg Sep It.	Avg CG It.	Avg Columns	Avg +rows (%)	Benders' cuts	Avg RMP time (%)	Avg PP time (%)	Avg Sep time (%)	Feasibility rate (%)
LNS (with Benders' cuts)	1	2.43	2.71	0.39	39.0	93.6	2.2	5.7	116.3	283.1	1.33	162	41	26	16	90
	1.5	2.52	2.79	0.42	48.3	76.9	2.3	5.9	127.5	315.3	1.36	129	38	34	13	92
	2	2.30	2.73	0.30	36.1	72.6	2.1	5.4	121.5	311.5	1.17	117	34	39	13	92
	2.5	2.40	2.77	0.47	56.7	57.6	2.6	6.3	148.5	359.3	1.47	127	34	43	13	90
	3	2.53	2.80	0.30	36.4	50.3	2.9	7.0	154.0	402.7	1.55	124	36	44	12	92
LNS (without Benders' cuts)	1	2.57	2.84	0.46	46.3	162.1	0.6	3.9	99.4	237.3	1.15	0	44	22	5	93
	1.5	2.26	2.60	0.24	27.7	146.4	0.6	3.9	98.8	238.4	1.03	0	41	28	4	94
	2	2.42	2.69	0.24	29.1	129.6	0.7	3.7	102.3	248.6	1.07	0	38	34	4	92
	2.5	2.46	2.73	0.35	42.1	100.7	0.8	4.6	124.5	299.0	1.43	0	36	42	4	91
	3	2.30	2.68	0.29	34.5	100.9	0.6	3.9	109.9	268.3	1.14	0	33	45	4	93
Random Iterative	1	2.93	3.07	0.52	52.2	34.4	7.0	23.2	207.4	746.1	11.57	0	66	25	6	49
	1.5	2.64	2.90	0.29	32.9	27.5	6.9	22.6	229.9	779.7	11.87	0	62	31	4	43
	2	2.79	2.97	0.30	36.1	19.2	7.0	23.8	256.8	908.0	11.97	0	73	23	2	53
	2.5	2.71	2.94	0.27	33.0	18.7	7.2	25.2	298.1	993.2	10.37	0	77	20	2	66
	3	2.76	3.05	0.34	41.5	14.5	6.8	23.3	289.4	1000.9	13.70	0	70	27	2	50

Table 6.8: Average solution quality over all instances

	LNS (with Benders' cuts)	LNS (without Benders' cuts)	Random iterative	LNS (with paths' length as acceptance criterion)
Passenger travel time	2.88 %	2.81 %	3.15 %	3.24 %
Paths' length	0.57 %	0.45 %	0.62 %	0.33 %

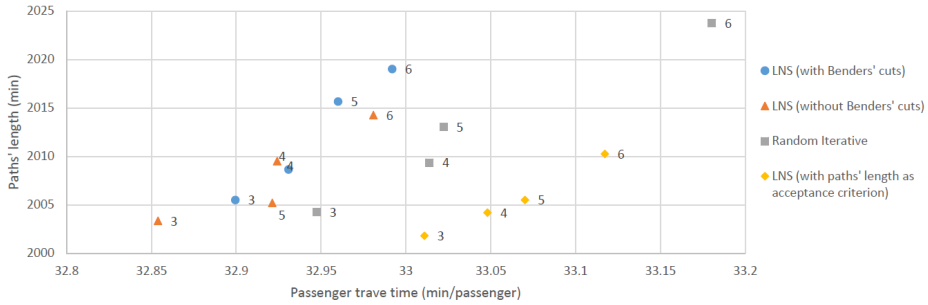


Figure 6.12: Average solution values instances with different values of HW_k in minutes.

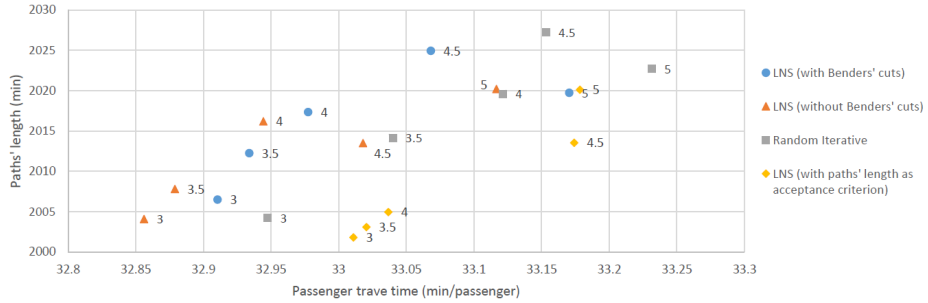


Figure 6.13: Average solution values instances with different values of HW_n in minutes.

6.5.2.3 Effect of Benders' cuts

The addition of a limited amount of Benders' cuts within the presented method does not have a significant impact. Additional tests were carried out where the time limit is extended so that more Benders' cuts can be generated. Nevertheless, the solution method does not provide better solutions. In fact, the quality decreases. It is observed that both generating more cuts or using a larger value of α, increases the fractionality of the solutions and the complexity of the RMP. Furthermore, solutions with a high number of columns with small coefficients hinder the dive heuristic procedure.

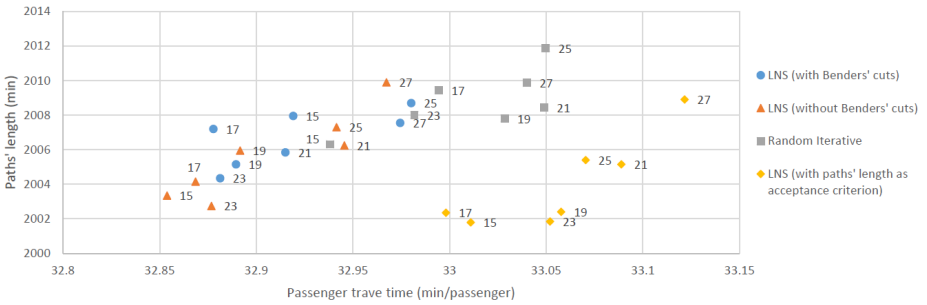


Figure 6.14: Average solution values instances with different values of HW_s in minutes.

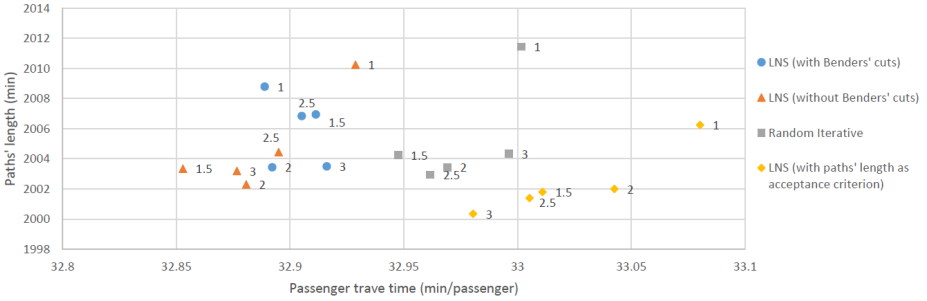


Figure 6.15: Average solution values instances with different values of κ in $\pm$ minutes.

Alternatively, in order to assess the potential impact of the Bender cuts, we solve the root node adding all violated Benders' cuts and compare the solution value with the manual lower bound mentioned at the end of Section 6.4.6. For this experiment, the objective function is modified such that only passenger travel time is considered. This is tested on the instance with base parameter values considering the network with station clusters. The results are summarized in Table 6.9 where the first column shows the optimality gap from the precomputed manual lower bound to the best known integer solution. The second column shows the optimality gap from the root node solution solved. The number of cuts and computational time is shown in the last two columns respectively. The results indicate that the Benders' cuts are able to provide a stronger lower bound. It should be noted that solving the entire branch-and-bound tree adding all the necessary violated Benders' cuts guarantees converging to the optimal solution. This suggest that addition of Benders' cuts may be more interesting in an exact method for the integrated model compared to in a heuristic as proposed here. It is clear, however, that such an exact method only could solve much smaller instances compared to those considered in this paper.

Table 6.9: Lower bound (LB) comparison for the base case instance with cluster stations.

Manual lower bound	LB with Benders' cuts	Benders' cuts	Time (s)
0.72 %	0.61 %	926	5885

6.6 Conclusion

As future work, it would be interesting to study how to decrease the time for solving the LP relaxation of the master problem as this can become quite excessive for the more constrained instances. One may attempt to 1) leave the headway constraints out of the initial formulation and add the violated ones by separation or 2) attempt to stabilize the dual variables in the column generation algorithm (see e.g., Du Merle et al. (1999) or Oukil et al. (2007)).

Computational results showed that the addition of Benders' cuts in the LNS did not improve the performance of the heuristic even though the cuts allow us to fully integrate the passenger travel time objective in the mathematical model that is the foundation of the LNS heuristic. It is possible that further work could change this conclusion. If the time for solving the passenger flow sub-problem could be reduced and convergence of the Benders' algorithm could be improved, such that fewer Benders' cuts are needed then the approach may be more competitive. Improved convergence of the Benders' algorithm could perhaps be achieved using generic speed-up techniques as those suggested in Magnanti and Wong (1981), Papadakos (2008) and Fischetti et al. (2017).

When looking at the number of columns needed per iteration, it is also interesting to look from which lines the columns mainly come from. In average, 93% of all the columns generated belong to lines using the quadruple-track segment. Allowing two routes for the trains doubles the number of possible columns that can be generated. It should also be noted that 32 % of the total amount of columns belong to the two lines running until Kalundborg. This is related to the fact that at the single-track segment of this corridor, is the only place where a crossing between trains of different lines can occur. In order to cross, one of the trains needs to dwell for three minutes in one of the stations resulting in a poor path length. As the crossing constraints are added by separation, this results in a larger amount of columns generated.

The model is able to route the passengers realistically. This is analyzed using graphical tools such as the one shown in Figure 6.17 which shows the passenger flow between trains at København H for an example solution. Nevertheless, a more realistic routing of the passengers in the most congested areas can help to have a complete perspective of the trips of the passengers and the occupancy

of the trains. This can be further improved by taking train capacity into account (Rezanova, 2015) or achieving a more accurate estimation of the passenger demand.

Although the fixed running times between stations simulate realistic cases to a large extent, considering variable running times at the track segments can increase significantly the solution space. However, the complexity of the model would increase accordingly. Also, considering different types of headway along the network allows a better utilization of the track capacity as more trains can be scheduled per corridor (Liu and Han, 2017).

Different graphical tools have been used to analyze the potential additional conflicts of a timetable such as the one in Figure 6.16 which shows an example graphic timetable for the north-western corridor between København H and Kalundborg. Routing the trains at a more detailed level at some stations can allow having completely conflict-free solutions in the network. Currently, feasibility issues may arise from the model due to track-crossing conflicts at some stations where corridors join. This can be solved by adding additional graph nodes to model the track junctions. Likewise, turnaround times for trains at the end of stations can be enforced by removing the conflicting arcs in the graph. This can potentially lead to a better utilization of the rolling stock.

In conclusion, this paper aims at optimizing the railway timetable generation process from a passenger perspective. Solution methods have been implemented to solve the network for Regional and InterCity trains in Zealand. The methods are based on a graph formulation that takes advantage of the symmetric timetabling strategy and the assumed fixed train running times between stations. As a result, all the required train paths for a line in a cycle time of one hour can be computed by a single *shortest path*. Furthermore, the algorithms rely mainly on both column generation and constraint separation techniques. This, combined with Benders' cuts that guide the routing of the passengers results in an algorithm for railway timetabling that optimizes passenger travel time. The methods have been shown to find good solutions to the network in a relatively fast time. The minimum headway can be easily increased along the network, achieving more robust timetables, without a significant detriment in time or solution quality.

Last but not least, the graph representation of the problem has the potential to easily model parts of the network in more detail such as track-crossing conflicts or platform assignment. The methods can potentially be improved and implemented as a useful tool in the planning process of a train operating company.

Acknowledgement: The work of Stefan Ropke was funded by the Innovation Fund Denmark under the IPTOP project, this support is gratefully acknowl-

edged. The authors are grateful to Federico Farina for his valuable comments and useful discussions. Moreover, thanks are due to Esben Linde from DSB for having provided relevant data for the instances presented in the paper. Last but not least, the authors are thankful to the Editor Prof. Oliveira and to five anonymous reviewers for their constructive remarks that have helped improve the manuscript.

References

Barrena, E., Canca, D., Coelho, L. C., and Laporte, G. (2014). Single-line rail rapid transit timetabling under dynamic passenger demand. *Transportation Research Part B: Methodological*, 70:134–150.

Bezanson, J., Edelman, A., Karpinski, S., and Shah, V. B. (2017). Julia: A fresh approach to numerical computing. *SIAM review*, 59(1):65–98.

Borndörfer, R., Hoppmann, H., and Karbstein, M. (2017). Passenger routing for periodic timetable optimization. *Public Transport*, 9(1-2):115–135.

Brannlund, U., Lindberg, P., Nou, A., and Nilsson, J.-E. (1998). Railway timetabling using Lagrangian relaxation. *Transportation Science*, 32(4):358–369.

Bussieck, M. R., Winter, T., and Zimmermann, U. T. (1997). Discrete optimization in public rail transport. *Mathematical Programming*, 79(1):415–444.

Cacchiani, V., Caprara, A., and Toth, P. (2008). A column generation approach to train timetabling on a corridor. *4OR*, 6(2):125–142.

Cacchiani, V., Caprara, A., and Toth, P. (2010a). Non-cyclic train timetabling and comparability graphs. *Operations Research Letters*, 38(3):179–184.

Cacchiani, V., Caprara, A., and Toth, P. (2010b). Scheduling extra freight trains on railway networks. *Transportation Research Part B: Methodological*, 44(2):215–231.

Cacchiani, V., Caprara, A., and Toth, P. (2013). Finding cliques of maximum weight on a generalization of permutation graphs. *Optimization Letters*, 7(2):289–296.

Cacchiani, V. and Toth, P. (2012). Nominal and robust train timetabling problems. *European Journal of Operational Research*, 219(3):727–737.

Caprara, A., Fischetti, M., and Toth, P. (2002). Modeling and solving the train timetabling problem. *Operations Research*, 50(5):851–861.

Caprara, A., Kroon, L., Monaci, M., Peeters, M., and Toth, P. (2007). Chapter 3 passenger railway optimization. In Barnhart, C. and Laporte, G., editors, *Transportation*, volume 14 of *Handbooks in Operations Research and Management Science*, pages 129–187. Elsevier.

Caprara, A., Monaci, M., Paolo Toth, P., and Guida, P. L. (2006). A Lagrangian heuristic algorithm for a real-world train timetabling problem. *Discrete Applied Mathematics*, 154(5):738–753. IV ALIO/EURO Workshop on Applied Combinatorial Optimization.

Carey, M. (1994). Extending a train pathing model from one-way to two-way track. *Transportation Research, Part B: Methodological*, 28B(5):395–400.

Carey, M. and Lockwood, D. (1995). A model, algorithms and strategy for train pathing. *Journal of the Operational Research Society*, 46(8):988–1005.

Cordeau, J., Toth, P., and Vigo, D. (1998). A survey of optimization models for train routing and scheduling. *Transportation Science*, 32(4):380–404.

DSB (2018). Langsigtet Planlægning.

Du Merle, O., Villeneuve, D., Desrosiers, J., and Hansen, P. (1999). Stabilized column generation. *Discrete Mathematics*, 194(1-3):229–237.

Farina, F. (2019). *Optimization of operations in public transportation.* PhD thesis, Technical University of Denmark.

Fischer, F. (2015). Ordering constraints in time expanded networks for train timetabling problems. *Openaccess Series in Informatics*, 48:97–110.

Fischer, F. and Schlechte, T. (2017). Strong relaxations for the train timetabling problem using connected configurations. *Openaccess Series in Informatics*, 59.

Fischetti, M., Ljubic, I., and Sinnl, M. (2017). Redesigning benders decomposition for large-scale facility location. *Management Science*, 63(7):2146–2162.

Gattermann, P., Großmann, P., Nachtigall, K., and Schöbel, A. (2016). Integrating passengers' routes in periodic timetabling: A sat approach. In *16th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS 2016)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

Hansen, I. A. (2009). Railway network timetabling and dynamic traffic management. In *2nd International Conference on Recent Advances in Railway Engineering, ICRARE-2009, Tehran, IR Iran, Sept. 27-28, 2009.*

Harrod, S. (2012). A tutorial on fundamental model structures for railway timetable optimization. *Surveys in Operations Research and Management Science*, 17(2):85–96.

Jovanovic, D. and Harker, P. T. (1991). Tactical scheduling of rail operations - the SCAN-I system. *Transportation Science*, 25(1):46–64.

Kinder, M. (2008). Models for periodic timetabling. Master's thesis, Technische Universität, Berlin.

Li, D. W., Ding, S. S., Zhang, Q., and Li, S. (2017). Improved dynamic demand oriented timetabling model for intercity railway. *Jiaotong Yunshu Xitong Gongcheng Yu Xinxi/journal of Transportation Systems Engineering and Information Technology*, 17(3):157–164.

Liebchen, C. (2007). Periodic timetable optimization in public transport. In Waldmann, K.-H. and Stocker, U. M., editors, *Operations Research Proceedings 2006*, pages 29–36, Berlin, Heidelberg. Springer Berlin Heidelberg.

Liebchen, C. and Möhring, R. H. (2002). A case study in periodic timetabling. *Electronic Notes in Theoretical Computer Science*, 66(6):18–31.

Liu, P. and Han, B. (2017). Optimizing the train timetable with consideration of different kinds of headway time. *Journal of Algorithms & Computational Technology*, 11(2):148–162.

Lubin, M. and Dunning, I. (2015). Computing in operations research using julia. *INFORMS Journal on Computing*, 27(2):238–248.

Lusby, R. M., Larsen, J., Ehrgott, M., and Ryan, D. (2011). Railway track allocation: Models and methods. *OR Spectrum - Quantitative Approaches in Management*, 33(4):843–883.

Magnanti, T. L. and Wong, R. T. (1981). Accelerating benders decomposition: Algorithmic enhancement and model selection criteria. *Operations Research*, 29(3):464–484.

Martin-Iradi, B. (2018). Optimization in railway timetabling for regional and intercity trains in Zealand. Master's thesis, Technical University of Denmark.

Min, Y. H., Park, M. J., Hong, S. P., and Hong, S. H. (2011). An appraisal of a column-generation-based algorithm for centralized train-conflict resolution on a metropolitan railway network. *Transportation Research Part B: Methodological*, 45(2):409–429.

Nachtigall, K. (1998). Periodic network optimization and fixed interval timetables. *Deutsches Zentrum für Luft–und Raumfahrt, Institut für Flugführung, Braunschweig*.

Odijk, M. A. (1996). A constraint generation algorithm for the construction of periodic railway timetables. *Transportation Research Part B: Methodological*, 30(6):455–464.

Oukil, A., Amor, H. B., Desrosiers, J., and El Gueddari, H. (2007). Stabilized column generation for highly degenerate multiple-depot vehicle scheduling problems. *Computers & Operations Research*, 34(3):817–834.

Papadakos, N. (2008). Practical enhancements to the magnanti-wong method. *Operations Research Letters*, 36(4):444–449.

Peeters, L. (2003). *Cyclic railway timetable optimization.* PhD thesis, Erasmus University of Rotterdam.

Polinder, G.-J., Cacchiani, V., Schmidt, M., and Huisman, D. (2020). An iterative heuristic for passenger-centric train timetabling with integrated adaption times. Technical Report ERS-2020-006-LIS.

Rezanova, N. J. (2015). Line planning optimization at DSB. In *13th Conference on advanced systems in public transport, Erasmus University.*

Ropke, S. and Pisinger, D. (2006). An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation Science*, 40(4):455–472.

Schöbel, A. and Scholl, S. (2006). Line Planning with Minimal Traveling Time. In Kroon, L. G. and Möhring, R. H., editors, *5th Workshop on Algorithmic Methods and Models for Optimization of Railways (ATMOS'05)*, volume 2 of *OpenAccess Series in Informatics (OASIcs)*, Dagstuhl, Germany. Schloss Dagstuhl–Leibniz-Zentrum für Informatik.

Serafini, P. and Ukovich, W. (1989). A mathematical model for periodic scheduling problems. *SIAM J. Discret. Math.*, 2(4):550–581.

Shaw, P. (1998). Using constraint programming and local search methods to solve vehicle routing problems. *Principles and Practice of Constraint Programming - Cp98. 4th International Conference, Cp98. Proceedings*, pages 417–31.

Szpigel, B. (1973). Optimal train scheduling on a single track railway. *Proceedings of the IFORS Conference Operational Research '72*, pages 343–352.

Zhang, Y., Peng, Q., Yao, Y., Zhang, X., and Zhou, X. (2019). Solving cyclic train timetabling problem through model reformulation: Extended time-space network construct and alternating direction method of multipliers methods. *Transportation Research Part B: Methodological*, 128:344–379.

Zhou, W., Fan, W., You, X., and Deng, L. (2019). Demand-oriented train timetabling integrated with passenger train-booking decisions. *Sustainability*, 11(18):4932.

Zhou, W., Tian, J., Xue, L., Jiang, M., Deng, L., and Qin, J. (2017). Multi-periodic train timetabling using a period-type-based lagrangian relaxation decomposition. *Transportation Research Part B: Methodological*, 105:144–173.

6.A Appendix

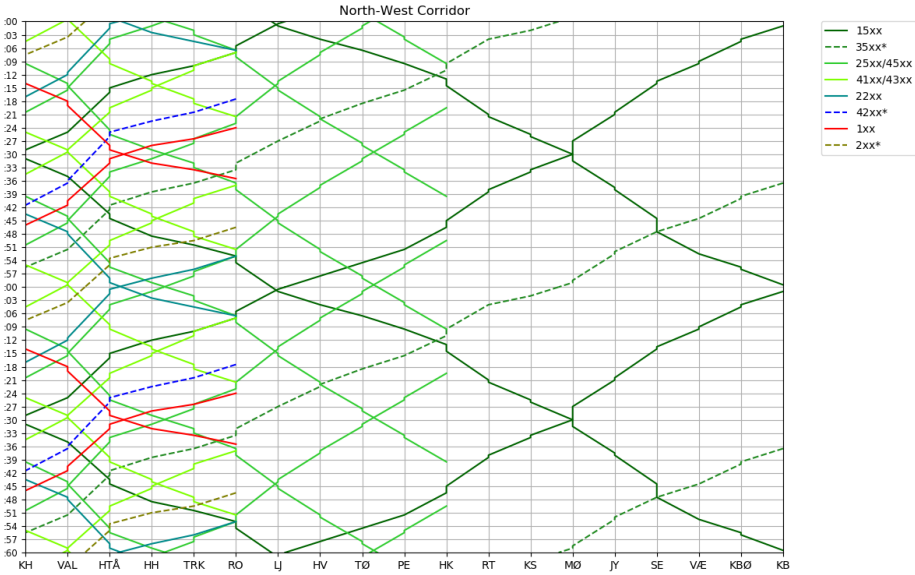


Figure 6.16: Timetable example for the lines running through the North-West corridor

Table 6.10: Average performance of LNS method with paths' length as acceptance criterion.

HW_k (min)	Best PTT gap (%)	PTT gap (%)	PL gap (%)	Var. PL gap (%)	Alg It.	Avg Dive It.	Avg Sep It.	Avg CG It.	Avg Columns	Avg +rows (%)	Benders cuts	Avg RMP time (%)	Avg PP time (%)	Avg Sep time (%)	Feasibility rate (%)
3	2.59	3.10	0.17	18.86	143.7	0.6	3.7	97.7	242.7	1.01	0	40	29	4	93
4	2.87	3.21	0.29	32.57	146.5	0.6	3.5	96.8	251.1	0.83	0	40	29	4	92
5	2.84	3.28	0.35	40.00	166.5	0.5	3.1	76.4	225.8	0.60	0	37	28	4	93
6	3.08	3.43	0.59	67.14	167.0	0.5	2.9	75.5	236.2	0.48	0	39	27	4	91
HW_n (min)	Best PTT gap (%)	PTT gap (%)	PL gap (%)	Var. PL gap (%)	Alg It.	Avg Dive It.	Avg Sep It.	Avg CG It.	Avg Columns	Avg +rows (%)	Benders cuts	Avg RMP time (%)	Avg PP time (%)	Avg Sep time (%)	Feasibility rate (%)
3	2.59	3.10	0.17	18.9	144.1	0.6	3.7	98.0	243.5	1.01	0	40	29	4	93
3.5	2.71	3.13	0.23	26.3	148.3	0.5	3.2	92.9	248.8	0.93	0	40	29	4	92
4	2.72	3.18	0.32	36.9	143.5	0.4	2.9	87.3	253.9	0.80	0	40	29	4	94
4.5	3.04	3.61	0.75	86.0	89.1	0.7	4.3	124.7	417.6	2.35	0	58	28	3	87
5	3.05	3.62	1.08	123.5	51.0	1.9	9.8	311.7	1240.4	7.70	0	91	7	1	71
HW_s (min)	Best PTT gap (%)	PTT gap (%)	PL gap (%)	Var. PL gap (%)	Alg It.	Avg Dive It.	Avg Sep It.	Avg CG It.	Avg Columns	Avg +rows (%)	Benders cuts	Avg RMP time (%)	Avg PP time (%)	Avg Sep time (%)	Feasibility rate (%)
15	2.59	3.10	0.17	18.9	140.8	0.6	3.7	97.7	242.8	1.01	0	40	29	4	93
17	2.57	3.06	0.19	22.0	141.2	0.6	3.9	95.9	228.9	1.20	0	39	29	5	92
19	2.89	3.24	0.20	22.3	137.3	0.5	3.5	100.6	247.5	1.37	0	40	29	5	92
21	2.98	3.34	0.33	38.0	127.8	0.8	4.5	106.9	268.4	2.01	0	40	32	6	87
23	2.75	3.22	0.17	19.1	147.8	0.5	3.3	89.1	221.3	1.44	0	37	28	7	93
25	3.09	3.28	0.35	39.4	128.6	0.5	3.7	106.5	260.0	2.06	0	39	30	8	90
27	2.79	3.44	0.52	59.4	124.6	0.6	3.8	101.1	251.1	2.49	0	38	30	9	88
κ ($\pm$ min)	Best PTT gap (%)	PTT gap (%)	PL gap (%)	Var. PL gap (%)	Alg It.	Avg Dive It.	Avg Sep It.	Avg CG It.	Avg Columns	Avg +rows (%)	Benders cuts	Avg RMP time (%)	Avg PP time (%)	Avg Sep time (%)	Feasibility rate (%)
1	2.68	3.31	0.26	26.2	158.7	0.6	3.8	96.8	230.4	1.11	0	43	22	5	92
1.5	2.59	3.10	0.17	18.9	140.0	0.6	3.7	98.2	243.6	1.01	0	40	29	4	93
2	2.85	3.19	0.23	27.3	128.4	0.6	3.6	98.8	241.8	0.99	0	37	35	4	93
2.5	2.75	3.08	0.20	23.6	107.4	0.8	4.3	107.5	263.5	1.30	0	34	41	4	93
3	2.34	3.00	0.14	17.3	107.0	0.6	3.7	98.9	255.0	1.01	0	31	44	4	93

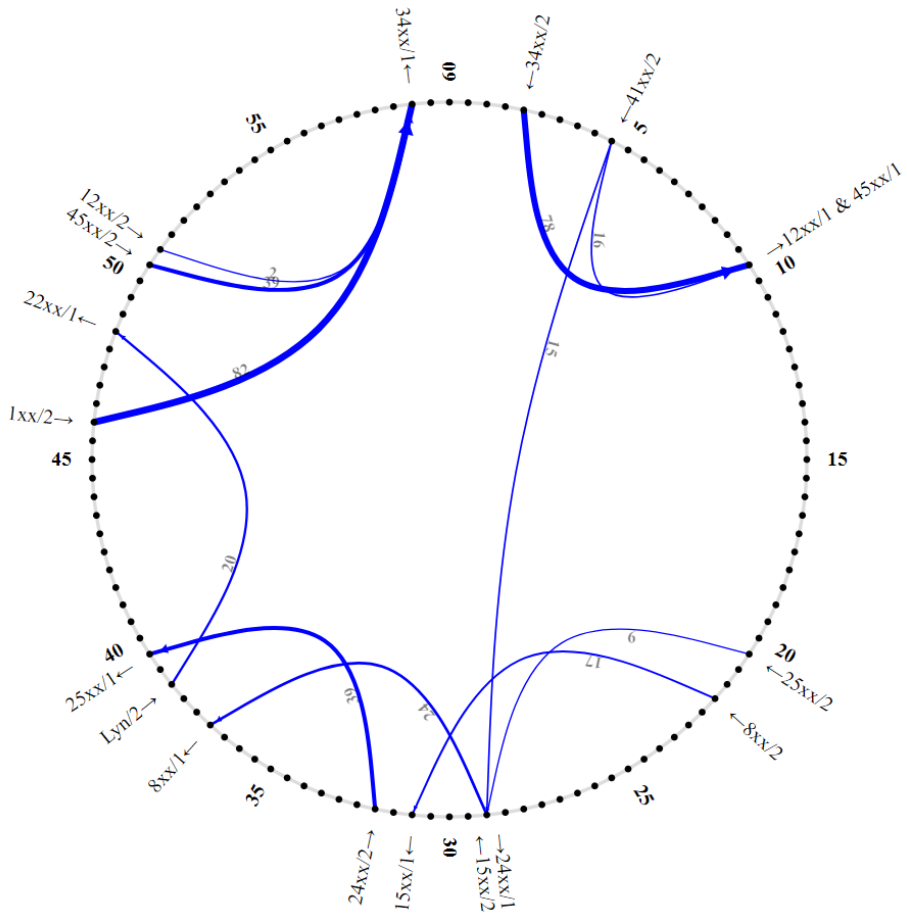


Figure 6.17: Example diagram of amount of passengers transferring between trains at København H during a rush hour