Transportation Research Part E 168 (2022) 102922

Contents lists available at ScienceDirect

Transportation Research Part E

journal homepage: www.elsevier.com/locate/tre

Drayage routing with heterogeneous fleet, compatibility constraints, and truck load configurations

Daniel Bustos-Coral *, Alysson M. Costa

School of Mathematics and Statistics, The University of Melbourne, VIC 3010, Australia

ARTICLE INFO

Keywords:
Drayage routing
Heterogeneous fleet
Compatibility constraint
Truck load configuration
Mixed-integer programming

ABSTRACT

This paper addresses drayage routing problems with heterogeneous fleets, compatibility constraints, and truck load configurations. In these problems, containers can be of any size and cargo category. In addition, compatibility and load configuration constraints define which and how many containers can be transported by each truck. We propose a state transition logic to model these constraints. Based on this logic, we develop two mixed-integer programming models for this family of problems. The first model is a compact formulation that can be input into a black-box solver. The second model combines partial routes in a space–time network and is solved with a tailored branch-and-cut approach. To analyze the efficiency of the proposed models, we conduct extensive computational tests on instances with different numbers of requests, geographical distributions of locations, time-window lengths, and fleet compositions. Moreover, we discuss how our modeling approach can assist decision-making in three different drayage routing applications.

1. Introduction

The transportation of containerized cargo across intermodal networks is carried out in three stages. First, in the *pre-haulage stage*, containers are trucked from shippers' locations to an origin terminal, where they are consolidated. Next, in the *main haulage stage*, long-haul vehicles (typically ships or trains) move the containers from the origin terminal to a destination terminal. This stage may involve transshipments, where the containers are transferred from one long-haul vehicle to another to continue their journey to the destination terminal. Finally, in the *post-haulage stage*, the containers are trucked from the destination terminal to their final destinations.

Drayage — which refers to the transport of containers by trucks — is a key aspect of intermodal operations. In the pre- and post-haulage stages, drayage enables the transfer of shipments to and from long-haul vehicles, respectively. In the main-haulage stage, drayage can occur as part of transshipment processes, e.g., when trucks move containers between rail ramps.

As Chen et al. (2022) note, the overall performance of the container supply chain largely depends on the performance of drayage operations that, in turn, can have major economic, environmental, and urban impacts. In consequence, drayage has increasingly drawn the attention of researchers of different areas, such as transportation, operations research, civil engineering, and economics.

From the disciplines that have studied drayage, operations research stands out as one that has done so more intensively, as Chen et al. (2022) also note. In this regard, we can mention that a wide variety of operations research models have been proposed for drayage problems with different characteristics. The simplest (and most studied) of these models consider a single type of container and a single type of truck. On the other hand, more complex models can consider aspects such as the following:

* Corresponding author.
E-mail addresses: dbustos@student.unimelb.edu.au (D. Bustos-Coral), alysson.costa@unimelb.edu.au (A.M. Costa).

https://doi.org/10.1016/j.tre.2022.102922
Received 25 October 2021; Received in revised form 21 September 2022; Accepted 27 September 2022
Available online 4 November 2022
1366-5545/© 2022 Elsevier Ltd. All rights reserved.

- **Heterogeneous containers:** Containers can have different specifications regarding size and type of cargo, e.g., 53ft dry, 40ft reefer, and 20ft tank containers.
- **Compatibility constraints:** Trucks in the fleet may be suited to haul specific types of containers, depending on their chassis, permits, or additional equipment, e.g., some trucks may have a chassis suitable to carry 48ft and 53ft containers, some trucks may be equipped with a generator set to power reefer containers, and some trucks may have a permit to carry out-of-gauge (OOG) loads. In addition, there can be compatibility constraints between containers depending on their loads, e.g., containers carrying dangerous goods should not be moved alongside other containers.
- **Truck load configurations:** Trucks in the fleet can allow different load configurations that determine their capacity for carrying containers of different sizes and with different loads, e.g., a truck with a 40ft chassis can haul one (empty or full) 40ft container, one full 20ft container, or up to two empty 20ft containers.

As part of this study, we present a literature review that classifies a wide variety of drayage routing problems according to the above characteristics, namely: the homogeneity/heterogeneity of the containers, the consideration of compatibility constraints, and the consideration of truck load configurations. This review identifies a research gap: no generic model allows considering arbitrary container types, trucks with specific load configurations, and compatibility constraints regarding container size and cargo. Aiming to bridge this gap, we address a drayage routing problem focused on the characterization of trucks and containers. This problem is flexible enough to consider any container type and associated constraints. We call this problem the *drayage routing problem with heterogeneous fleet, compatibility constraints, and truck load configurations* (DRP-HCL).

The modeling of the DRP-HCL is based on two observations. First, service requests in drayage problems can be represented by origin–destination pairs indicating where each container should be picked up and where it should be delivered, often with associated time-window constraints. In this sense, drayage problems are related to the *pickup and delivery problem* (PDP; Berbeglia et al., 2007). Second, drayage problems often involve decisions about fleet acquisition, e.g., when container chassis must be rented. So, these problems can take elements of the *fleet size and mix vehicle-routing problem* (FSMVRP; Golden et al., 1984).

To model the trucks' load capacities and compatibility constraints, we propose a state transition logic that represents the possible load configurations of a truck along its route. Based on this logic, we develop two mixed-integer programming (MIP) formulations for the DRP-HCL.

The first formulation uses the proposed state transition logic to assign variables that represent the state of a truck at each stop along its route. The main feature of this model is the use of logical constraints that link variables representing truck states to arc-traversal variables that define the routes traveled by the trucks. Because this model is compact, it can be solved directly with the use of a black-box MIP solver.

The second formulation, aimed at solving larger instances, builds upon the work of Alyasiry et al. (2019); it consists of using the state transition logic to generate feasible route fragments. Then, timed versions of these fragments are connected in a space–time network to obtain feasible routes. This formulation is solved by a branch-and-cut algorithm that uses lazy constraints to remove subtours and time-window infeasible paths.

We use both formulations to solve a large set of instances with different numbers of requests, geographical distributions of locations, truck and container fleets, and time-window settings. We thoroughly discuss the influence of the instance characteristics on solving times, commenting on the limitations of the formulations. In addition, we consider three realistic situations arising in drayage operations where we use the DRP-HCL as a framework to gain managerial insights.

The rest of this paper is organized as follows: Section 2 presents the literature review. Section 3 defines the DRP-HCL. Section 4 presents the formulations of the DRP-HCL. First, Section 4.1 describes the state transition logic we use to model the trucks' load capacities and compatibility constraints. Then, Sections 4.2 and 4.3 present the two proposed MIP formulations. Section 5 discusses computational results. In particular, Section 5.2 describes the set of instances. Sections 5.4 and 5.5 present the obtained results. Section 5.6 analyzes the performance of the proposed models in view of the instances' characteristics. Section 5.7 addresses the three situations where we use the DRP-HCL as a tool to gain managerial insights. Finally, Section 6 concludes the paper.

2. Related work

This section reviews studies on drayage problems, classifying them according to the types of containers they consider (homogeneous/heterogeneous) and the presence or absence of compatibility constraints and truck load configurations.

Table 1 presents the full list of articles reviewed, grouped according to the proposed classification. The table also presents information on the consideration of time-window constraints, truck types (homogeneous/heterogeneous), the type of solution approach proposed by the authors (exact/heuristic), and the size of the largest problem instance addressed. Further details on each paper are discussed in Sections 2.1 to 2.6, followed by a brief discussion positioning this study within the existing literature in Section 2.7.

2.1. Homogeneous containers/cargoes without truck load configurations

Most of the articles in this section focus on aspects of drayage different from fleet characterization. Thus, trucks and containers are not characterized in detail.

Many of the problems reviewed in this section involve the repositioning of empty containers/equipment. This issue involves selecting the origin or destination of a load and occurs when, for example, a customer requires an empty container to pack their

Table 1
Comparison of drayage problems based on fleet characterization.

Paper	Het. cont./cargo	Container/cargo types	Load conf.	Het. trucks	Comp. constr.	Time windows	Solution approach	Size of largest instance
Section 2.1								
Wang and Regan (2002)	✗	Unspecified loads	✗	✗	✗	✓	E/H	150 loads
Gronalt et al. (2003)	✗	Full truckloads	✗	✗	✗	✓	E/H	512 orders
Jula et al. (2005)	✗	Unspecified containers	✗	✗	✗	✓	E/H	100 nodes
Ileri et al. (2006)	✗	Trailers (l&e)	✗	✓	✗	✓	E/H	130 orders
Smilowitz (2006)	✗	Trailers and containers	✗	✗	✗	✓	E/H	1638 tasks
Francis et al. (2007)	✗	Trailers and containers	✗	✗	✗	✓	H	200 tasks
Imai et al. (2007)	✗	Unspecified containers	✗	✓	✗	✗	E/H	200 nodes (100 orders)
Namboothiri and Erera (2008)	✗	Unspecified containers	✗	✗	✗	✓	H	100 requests
Caris and Janssens (2009)	✗	Unspecified containers	✗	✗	✗	✓	E/H	200 customers
Zhang et al. (2009)	✗	Unspecified containers (f&e)	✗	✗	✗	✓	E/H	200 containers
Zhang et al. (2010)	✗	40ft containers (f&e)	✗	✗	✗	✓	E/H	75 loads
Máhr et al. (2010)	✗	Reefer containers	✗	✗	✗	✓	H	65 jobs a day, 33 days
Zhang et al. (2011)	✗	40ft containers (f&e)	✗	✗	✗	✓	E/H	75 containers
Nossack and Pesch (2013)	✗	Unspecified containers (f&e)	✗	✗	✗	✓	E/H	75 requests
Wang and Yun (2013)	✗	40ft containers (f&e)	✗	✗	✗	✓	E/H	40 containers
Sterzik and Kopfer (2013)	✗	40ft containers (f&e)	✗	✗	✗	✓	E/H	75 requests
Braekers et al. (2013)	✗	Unspecified containers (f&e)	✗	✗	✗	✓	E/H	200 nodes
Zhang et al. (2014)	✗	Unspecified containers (f&e)	✗	✗	✗	✓	E/H	75 orders
Braekers et al. (2014)	✗	Unspecified containers (f&e)	✗	✗	✗	✓	E/H	400 nodes
Xue et al. (2014)	✗	Unspecified containers (f&e)	✗	✗	✗	✓	E/H	100 pickups/ 100 deliveries
Xue et al. (2015)	✗	Unspecified containers (f&e)	✗	✗	✗	✓	E/H	100 pickups/ 100 deliveries
Shiri and Huynh (2016)	✗	40ft dry containers (f&e)	✗	✗	✗	✓	E/H	200 nodes
Schulte et al. (2017)	✗	Standard containers (f&e)	✗	✗	✗	✓	E	80 vertices
Song et al. (2017)	✗	Unspecified containers (f&e)	✗	✗	✗	✓	E	35 requests
Zhang et al. (2020)	✗	Unspecified containers (f&e)	✗	✗	✗	✓	E/H	80 tasks
Escudero-Santana et al. (2020)	✗	Unspecified containers (f&e)	✗	✗	✗	✓	E	200 orders
Section 2.2								
Lai et al. (2013)	✗	Unspecified containers (f&e)	✓	✓	✗	✗	E/H	50 customers
Lai et al. (2015)	✗	24.5ft containers (f&e)	✓	✗	✗	✗	E/H	50 customers
Daham et al. (2017)	✗	20ft containers (f&e)	✓	✓	✗	✓	E	350 orders
Ghezelsoflu et al. (2018)	✗	24.5ft containers (f&e)	✓	✓	✗	✓	E	50 customers/ 141 containers
Zhang et al. (2018)	✗	Foldable containers (f&e)	✓	✗	✗	✓	H	75 orders
You et al. (2020)	✗	Unspecified containers (f&e)	✓	✗	✗	✓	E/H	200 customers
Zhang et al. (2021)	✗	Dry containers (f&e), trailers	✓	✗	✗	✗	E/H	100 orders
Ghezelsoflu et al. (2021)	✗	20ft containers (f&e)	✓	✓	✗	✓	E	1000 customers/ 6000 requests
Xue et al. (2021)	✗	Unspecified containers (f&e)	✓	✗	✗	✗	E/H	100 customers
Section 2.3								
Reinhardt et al. (2016)	✓	Dry 20/40ft containers, normal and high cube (f&e)	✗	✗	✗	✓	E	308 orders
Section 2.4								
Vidović et al. (2011)	✓	20/40ft containers	✓	✗	✗	✗	E/H	76 requests
Popović et al. (2014)	✓	20/40ft containers	✓	✗	✗	✓	E/H	100 nodes
Zhang et al. (2015)	✓	20/40ft containers (f&e)	✓	✗	✗	✗	E/H	500 orders
Funke and Kopfer (2016)	✓	20/40ft containers (f&e)	✓	✗	✗	✓	E	11 requests
Vidović et al. (2017)	✓	20/40ft containers	✓	✗	✗	✓	E/H	193 tasks
Popović et al. (2018)	✓	20/40ft containers	✓	✗	✗	✓	E	25 tasks
Bruglieri et al. (2021)	✓	20/40ft containers	✓	✗	✗	✗	E	200 customers
Chen et al. (2021b)	✓	Containers with different weights	✓	✗	✗	✓	E/H	50 requests
Bjelić et al. (2022)	✓	20/40ft containers (f&e)	✓	✗	✗	✓	E/H	96 tasks

(*continued on next page*)

Table 1 (*continued*).

Section 2.5								
Tan et al. (2006)	✓	20/40ft containers (f&e)	✗	✓	✓(s)	✓	H	132 tasks
Meisel and Kopfer (2014)	✓	Unspecified requests	✗	✓	✓(s&c)	✓	E/H	500 requests
Regnier-Coudert et al. (2016)	✓	General freight	✗	✓	✓(s&c)	✓	H	203 jobs
Pérez Rivera and Mes (2017)	✓	Any container (f&e)	✗	✓	✓(s&c)	✓	E/H	50 jobs
Ritzinger et al. (2017)	✓	Any container (f&e)	✗	✓	✓(s&c)	✓	H	50 orders
Shiri and Huynh (2018)	✓	20/40ft containers (f&e), chassis	✗	✓	✓(s)	✓	E/H	150 job nodes
Tilk et al. (2018)	✓	Loading units	✗	✓	✓(s&c)	✓	E	76 tasks
Heggen et al. (2019)	✓	Any container (f&e)	✗	✓	✓(s)	✓	H	Up to 470 requests
Cui et al. (2022)	✓	20/40ft containers (f&e)	✗	✓	✓(s)	✓	E/H	400 requests
Section 2.6								
Chung et al. (2007)	✓	20/40ft containers	✓	✓	✓(s)	✓	H	1093 orders
Bomboi (2020)	✓	20ft ordinary/special (f&e)	✓	✗	✓(c)	✗	E	60 customers
Moghaddam et al. (2020)	✓	20/40ft containers (f&e)	✓	✓	✓(s)	✓	E	89 requests
Yang and Daham (2020)	✓	20/40ft containers (f&e)	✓	✓	✓(s)	✓	E/H	500 orders
Yang et al. (2021)	✓	20/40ft containers (f&e)	✓	✓	✓(s)	✓	E/H	500 orders
Chen et al. (2021a)	✓	Any container	✓	✓	✓(s)	✗	E/H	300 requests
This paper	✓	Any container (f&e)	✓	✓	✓(s&c)	✓	E	1000 requests

(f&e): full and empty (l&e): loaded and empty (s): size (c): cargo (s&c): size and cargo E: exact H: heuristic.

cargo, and such a container can be delivered from a depot or from another customer's location. Alternatively, a customer may require the collection of an empty container, which can be moved to a depot or to another customer's location.

Problems addressing empty container/equipment repositioning are studied by Smilowitz (2006), Francis et al. (2007), Braekers et al. (2013), and Braekers et al. (2014). Meanwhile, Sterzik and Kopfer (2013), Nossack and Pesch (2013), and Shiri and Huynh (2016) study problems involving empty container repositioning and multiple depots or terminals. In particular, Shiri and Huynh (2016) complement the problem with the inclusion of a terminal appointment system. Escudero-Santana et al. (2020), in turn, address a problem involving empty container repositioning and soft time windows, which can be violated by incurring a penalty.

Máhr et al. (2010), Zhang et al. (2011), Wang and Yun (2013), and Xue et al. (2015) address problems in which containers are packed/unpacked as they are moved between different locations. In addition, these problems have the following distinctive features: The problem of Máhr et al. (2010) involves uncertain service times and job arrivals. The problem of Wang and Yun (2013) considers an intermodal network of trucks and trains. The problem of Zhang et al. (2011) considers limited empty containers. The problem of Xue et al. (2015) involves deciding if trucks must stay with containers during packing/unpacking operations.

Zhang et al. (2009, 2010, 2014, 2020), Song et al. (2017), and Xue et al. (2014) describe problems involving empty container repositioning and containers that are packed/unpacked as they are moved between different locations. In particular, the problem of Zhang et al. (2020) also involves a limited number of empty containers, while Zhang et al. (2010) and Zhang et al. (2014) consider several depots. Meanwhile, the problems of Song et al. (2017) and Xue et al. (2014) require deciding if trucks must stay with containers during packing/unpacking operations.

Some studies focus on drivers' working conditions. Jula et al. (2005) and Gronalt et al. (2003) consider a maximum working time for drivers. Moreover, in the problem of Gronalt et al. (2003), drivers are pre-assigned to specific depots. The possibility of hiring external contractors is considered by Imai et al. (2007) and Ileri et al. (2006). In particular, Ileri et al. (2006) focus on driver conditions and consider earliest and latest start times, starting and ending locations, maximum working hours, and minimum daily wages.

Namboothiri and Erera (2008) and Schulte et al. (2017) address synchronization issues arising in drayage operations. Namboothiri and Erera (2008) consider a problem involving a port access control system, while Schulte et al. (2017) study a problem involving a truck appointment system that enables collaboration between truckers to reduce costs in terms of distance and emissions.

Wang and Regan (2002) and Caris and Janssens (2009) address drayage PDPs with fleets characterized in low detail and no further constraints. Remarkably, Wang and Regan (2002) proposed a time window partitioning approach that has been the basis for other solution approaches, such as that of Zhang et al. (2010).

2.2. Homogeneous containers/cargoes and truck load configurations

All the articles in this section are characterized by considering movements of containers that are packed/unpacked as they are moved between different locations.

Ghezelsoflu et al. (2018), Lai et al. (2013), Daham et al. (2017), and Ghezelsoflu et al. (2021) consider fleets composed of single and double container trucks. Moreover, Daham et al. (2017) consider limits on working time and cargo weight, while Ghezelsoflu et al. (2021) consider a planning horizon of several days, customers with soft time windows, and port costs associated with containers moved ahead or behind schedule.

Lai et al. (2015) consider trucks that haul one, two, four, or six containers at a time. Zhang et al. (2018) describe a problem involving foldable containers, which can be folded down and stacked over one another, making it possible for a truck to carry two or more folded containers at once. This problem also considers empty container repositioning and trucks that can leave containers during packing/unpacking operations. Zhang et al. (2021) address a problem involving a fleet of semi-trucks and trailers. The semi-trucks can be coupled with one or two trailers, and each trailer can carry a single container. This problem also involves empty container repositioning and trucks that can leave containers during packing/unpacking operations.

You et al. (2020) and Xue et al. (2021) address problems where trucks operate under the so-called semi-autonomous truck platooning mode, a truck convoy formed by a crewed truck leading a group of wireless connected semi-autonomous trucks. The load capacity of a platoon depends on the number of trucks that form it. These problems consider driver wages and fuel-related costs. In addition, You et al. (2020) consider empty container repositioning and trucks that can leave containers during packing/unpacking operations.

2.3. Heterogeneous containers/cargoes without truck load configurations or compatibility constraints

The only article in this section is authored by Reinhardt et al. (2016), who address a problem involving the repositioning of empty containers of different types considering several terminals and depots. In this case, demands for containers of specific types can only be matched with supplies of compatible containers. Nevertheless, the problem does not explicitly state compatibility constraints between containers and trucks.

2.4. Heterogeneous containers/cargoes with truck load configurations and without compatibility constraints

Vidović et al. (2011), Popović et al. (2014), Funke and Kopfer (2016), Zhang et al. (2015), Popović et al. (2018), and Bruglieri et al. (2021) consider trucks that haul one 40ft container or up to two 20ft containers. Furthermore, Vidović et al. (2011) and Popović et al. (2014) assume that routes are limited to four stops. Zhang et al. (2015) and Funke and Kopfer (2016) consider working time limits, containers that are packed/unpacked as they are moved between different locations, and empty container repositioning. In addition, Funke and Kopfer (2016) assume that it is necessary to decide whether the truck that delivers a container must stay with it while it is being packed/unpacked. Bruglieri et al. (2021) consider working time limits and a planning horizon of several days. Popović et al. (2018) consider trucks and container handlers that must synchronize their routes so that their arrival times at customers' locations match.

Vidović et al. (2017) consider three types of fleets, composed of either standard trucks, which haul one full 20ft or 40ft container; European modular system (EMS) trucks, which haul up to two full 20ft containers; or long combination vehicles (LCVs), which haul up to two 40ft, four 20ft, or even three 40ft containers. Standard trucks can make up to two stops per route, EMS trucks can make up to four stops per route, and LCVs can make up to eight stops per route.

Chen et al. (2021b) address a problem involving autonomous trucks that move containers between two container terminals in Singapore. The trucks can form platoons, which can move several containers at a time. This problem considers containers of different weights and trucks with explicit fuel consumption. The speed and load of a truck determine its fuel consumption. However, a truck following the lead truck of a platoon can save fuel as it faces reduced air resistance.

Bjelić et al. (2022) consider a dynamic routing problem involving movements of 20/40ft containers between customer locations and a depot. In particular, this problem assumes that empty containers are delivered to the depot after being unpacked by customers. The fleet is composed of either 80ft or 40ft trucks. The sources of dynamism of the problem are the arrival of new service requests, the cancellation of requests, changes in travel times due to traffic, and changes in time windows or service times. The approach to the problem consists of solving static versions of it at different time instants; each of the static problems includes any information disclosed by solving time.

2.5. Heterogeneous containers/cargoes with compatibility constraints and without truck load configurations

Regnier-Coudert et al. (2016), Shiri and Huynh (2018), and Tan et al. (2006) consider fleets of company-owned and subcontracted vehicles. Regnier-Coudert et al. (2016) consider a fleet composed of semi-trucks (which can pull trailers), straight trucks, and vans. Only some vehicles and trailers are compatible with specific tasks. For example, large-sized cargoes require high-capacity vehicles, and dangerous loads require vehicles operated by drivers with specialized training. Shiri and Huynh (2018) assume that semi-trucks, chassis, and containers are available at different locations, and the containers can only be moved with compatible chassis. Tan et al. (2006) consider semi-trucks that pull trailers available at trailer exchange points. There are two types of trailers, type 20 and type 40, which are compatible with specific tasks. Furthermore, trucks can leave containers during packing/unpacking operations.

Ritzinger et al. (2017) consider different truck and trailer types; the trailers are compatible with specific requests. In this case, while a container is being packed/unpacked, a truck can either leave the trailer with the container, leave only the container keeping the trailer, or wait. The trucks must serve as many requests as possible.

Pérez Rivera and Mes (2017) and Heggen et al. (2019) address problems arising in intermodal networks. Pérez Rivera and Mes (2017) consider a problem arising in a synchromodal network where the objective is to minimize routing costs and costs associated with the use of terminal facilities. This problem involves a fleet of trucks compatible with specific tasks, working time limits, empty container repositioning, containers that are packed/unpacked as they are moved between different locations, and trucks that can leave containers during packing/unpacking operations. Heggen et al. (2019) tackle a problem involving an intermodal network

with two service regions connected by long-haul transportation. In each region, fleets of trucks move containers between customer locations and terminals. The trucks are compatible with containers of specific sizes. In addition, this problem assumes that containers are packed/unpacked as they are moved between different locations.

Meisel and Kopfer (2014) and Tilk et al. (2018) address the active–passive vehicle-routing problem (APVRP), a generic problem suitable to represent drayage routing. The APVRP involves passive vehicles, which can carry cargo but cannot move autonomously, and active vehicles, which can move passive vehicles. Each passive vehicle can carry one cargo unit at a time, and each active vehicle can move one passive vehicle at a time. The APVRP entails compatibility constraints because passive vehicles can carry only certain cargoes, and active vehicles can move only certain passive vehicles. In addition, an active vehicle can leave a passive vehicle that is being loaded/unloaded. In the context of drayage, the cargoes correspond to containers, the passive vehicles correspond to container chassis, and the active vehicles correspond to semi-trucks that pull the chassis.

Cui et al. (2022) tackle a problem involving pickups and deliveries of 20/40ft containers. The problem assumes a fleet of semi-trucks that pull two types of container chassis — one compatible with 20ft containers, the other with 40ft containers. Each semi-truck can pull a single chassis at a time, and each chassis can carry a single container at a time. The problem involves chassis depots, where semi-trucks and chassis can be coupled or uncoupled. Alternatively, semi-trucks and chassis can be coupled at some pickup locations and uncoupled at some delivery locations.

2.6. Heterogeneous containers/cargoes with compatibility constraints and truck load configurations

The articles in this section describe trucks and containers in great detail, according to the proposed classification.

Chung et al. (2007), Moghaddam et al. (2020), Yang and Daham (2020), and Yang et al. (2021) address problems involving 20/40ft containers. Chung et al. (2007) consider heterogeneous trucks that can either carry two 20ft containers (or one 40ft container), a single 20ft container, or a single 40ft container. The trucks can be company-owned or subcontracted and have limited daily operation time. Moghaddam et al. (2020) consider trucks with different load capacities varying from 20ft to 80ft. The trucks stay with the containers or leave them during packing/unpacking operations. The trucks must be routed such that they serve as many requests as possible. Yang and Daham (2020) describe a fleet composed of 20ft trucks — which carry one 20ft container — and 40ft trucks — which carry one 40ft container or up to two 20ft containers. The trucks must be routed considering drivers' working hours and the payload weight of the containers. Yang et al. (2021) consider a problem involving empty container repositioning and trucks that stay with the containers or leave them during packing/unpacking operations. The trucks are identical to those described by Yang and Daham (2020).

Bomboi (2020) considers only 20ft containers and trucks that can haul up to two containers at a time. Some containers carry special cargoes (e.g., flammable or very heavy goods) and must be moved individually. In addition, the trucks can stay with the containers or leave them during packing/unpacking operations.

Chen et al. (2021a) tackle a problem involving two types of service requests: inbound requests, which consist of moving containers from an origin terminal to customer locations, and outbound requests, which consist of moving containers from customer locations to a destination terminal. The containers can be of any size and there can be different classes of trucks, with specific load capacities. Some trucks may not be able to move certain containers because they do not have enough load capacity.

2.7. Discussion

A large share of the drayage literature does not delve into fleet characterization. Indeed, most of the reviewed articles consider problems with homogeneous containers or trucks. The focus of those articles is on other issues, such as empty container repositioning or deciding if trucks should leave containers during packing/unpacking operations.

Despite this fact, there is a recent tendency toward tackling drayage problems characterizing trucks and containers in more detail, with heterogeneous trucks and containers, compatibility constraints, and truck load configurations. However, some limitations still remain, especially regarding the problems that explicitly define the load configurations of trucks. In such problems, the container types are homogeneous or, if they are heterogeneous, they are commonly limited to 20ft and 40ft sizes. Furthermore, in problems that specify truck load configurations, compatibility constraints are either related to container size (e.g., a truck with a 20ft chassis cannot carry a 40ft container) or cargo (e.g., only specialized trucks can move hazardous containers, which may require special handling), but not both simultaneously.

We aim to bridge these modeling gaps by proposing models for a drayage routing problem involving heterogeneous containers (of any size and cargo category), heterogeneous trucks with detailed load configurations, and compatibility constraints regarding container size and cargo.

3. Problem definition

The DRP-HCL consists of routing a fleet of trucks that move containers between pickup and delivery locations. There can be different container types with different sizes and loads. Likewise, there can be different truck types, with specific fixed costs, which can represent purchase or renting costs; routing costs, which can represent per mile costs; and load capacities. The movement of containers is subject to compatibility constraints. It is possible that only certain trucks can handle certain containers. It is also possible that some containers cannot be moved together because their loads are incompatible. All container movements must be

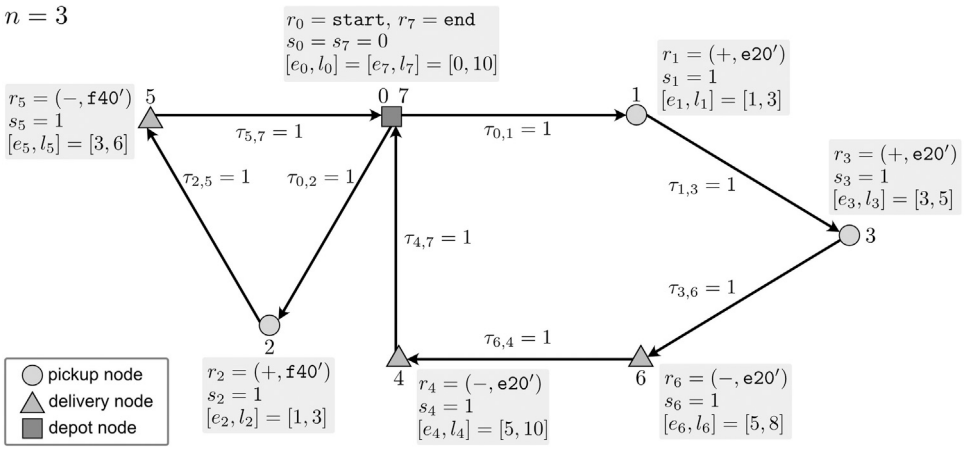


Fig. 1. A small example of routes defined on a graph $\mathcal{G}$.

performed within a planning horizon, considering that pickup and deliveries have separate time windows and service times. The aim of the problem is to plan the movement of a group of containers minimizing all costs.

Formally, we assume that there are n service requests. Let $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ be a directed graph with node set $\mathcal{N} = \{0, 1, \ldots, 2n, 2n+1\}$ and arc set $\mathcal{A} = \{(i, j) : i \in \mathcal{N}, j \in \mathcal{N}\}$. Nodes 0 and $2n + 1$ represent, respectively, the start and end depots for all trucks. $\mathcal{P} = \{1, \ldots, n\}$ is the subset of pickup nodes and $\mathcal{D} = \{n + 1, \ldots, 2n\}$ is the subset of delivery nodes. Each service request i, $1 \le i \le n$, is picked up at node $i \in \mathcal{P}$ and delivered at node $n + i \in \mathcal{D}$.

Each node is associated with a service description that specifies the service to perform at that node. Let $\mathcal{U} = \{+, -\}$ be the set of service types: + for pickup and − for delivery. Let $\mathcal{V}$ be the set of container types considered in a problem instance. For example, $\mathcal{V} = \{\texttt{e20}', \texttt{f20}', \texttt{e40}', \texttt{f40}', \texttt{e45}', \texttt{f45}'\}$ if the instance involves the movement of (empty or full) 20ft, 40ft, and 45ft dry containers. For each node $i \in \mathcal{P} \cup \mathcal{D}$, the service description is given by $r_i = (u_i, v_i) \in \mathcal{U} \times \mathcal{V}$. For each request j, $1 \le j \le n$, the service descriptions of its associated pickup and delivery nodes have the following pattern: $u_j = +$, $u_{n+j} = -$, and $v_j = v_{n+j}$; that is, the same container that is picked up at node j is delivered at node $n + j$. For example, if $r_j = (+, \texttt{e20}')$, then $r_{n+j} = (-, \texttt{e20}')$. For the nodes that represent the start and end depots, the service descriptions are given, respectively, by $r_0 = \texttt{start}$ and $r_{2n+1} = \texttt{end}$. The set of all service descriptions is given by $\mathcal{R} = (\mathcal{U} \times \mathcal{V}) \cup \{\texttt{start}, \texttt{end}\}$.

Each node $i \in \mathcal{N}$ has a service duration s_i and a time window $[e_i, l_i]$. For $i \in \mathcal{P} \cup \mathcal{D}$, $s_i > 0$ and e_i and l_i represent the earliest and latest times for starting service at the node, respectively. Meanwhile, the time windows of depot nodes, $[e_0, l_0] = [e_{2n+1}, l_{2n+1}]$, represent the planning horizon, and $s_0 = s_{2n+1} = 0$.

To characterize the trucks, we define a set of truck types $\mathcal{F} = \{1, \ldots, F\}$. The fixed cost associated with using a truck of type $f \in \mathcal{F}$ is given by E^f. The routing cost that a truck of type f incurs when it traverses an arc $(i, j) \in \mathcal{A}$ is given by $c^f_{i,j}$. Each truck type $f \in \mathcal{F}$ is compatible with specific containers and has a load capacity determined by the containers' sizes and loads. The travel time needed to traverse an arc $(i, j) \in \mathcal{A}$ is given by $\tau_{i,j}$, and is considered to be truck independent.

Fig. 1 illustrates a small example of routes defined on a graph $\mathcal{G}$. Nodes 0 and 7 represent, respectively, the start and end depots — both nodes are situated at the same location. The pickup nodes are given by $\mathcal{P} = \{1, 2, 3\}$ and the delivery nodes are given by $\mathcal{D} = \{4, 5, 6\}$. Empty 20ft containers are moved from node 1 to node 4 and from node 3 to node 6. A full 40ft container is moved from node 2 to node 5. The figure also shows travel times, time windows, and service times.

In the following section, we first model the trucks' load capacities and compatibility constraints. Then, we present two formulations to obtain optimal fleets and routes.

4. Formulations of the DRP-HCL

In this section, we present our two formulations for the DRP-HCL. Section 4.1 describes the state transition logic that models the trucks' load capacities and compatibility constraints. Then, Sections 4.2 and 4.3 describe two different approaches to determine fleet composition and truck routing.

4.1. State transition approach to model the trucks' load capacities and compatibility constraints

For each truck type $f \in \mathcal{F}$, we define a state transition graph composed of nodes (states) representing truck load configurations and arcs (state transitions) representing the transitions a truck makes between load configurations according to the services it performs along its route. We also define states and transitions that allow a truck to start and end its route at the depot.

Fig. 2 shows the state transition graph for heavy-duty trucks that can carry one full 20ft, 40ft, or 45ft dry container; one empty 20ft, 40ft, or 45ft dry container; or two empty 20ft dry containers. Nodes $1, \ldots, 8$ represent truck load configurations — e.g., node

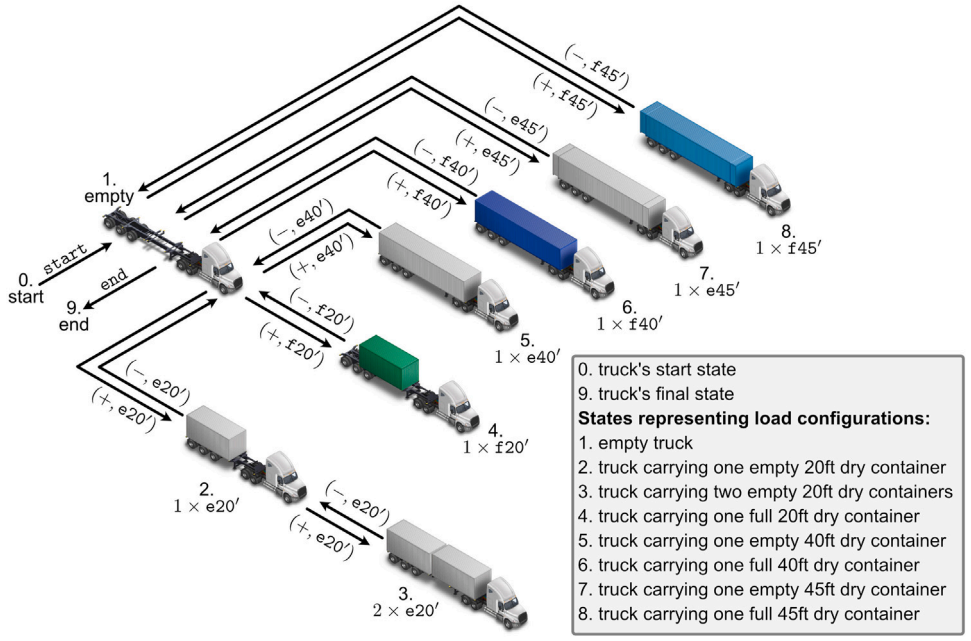


Fig. 2. State transition graph for heavy-duty trucks that haul 20ft, 40ft, and 45ft dry containers.

2, labeled as $1 \times \mathtt{e20}'$, indicates that the truck is carrying one empty 20ft dry container. Nodes 0 and 9 represent the truck's start and final states, respectively. Each arc is associated with a service description $r \in \mathcal{R}$. The arcs linking nodes $1, \ldots, 8$ represent the truck's transitions between load configurations — e.g., arc $(1, 8)$ represents the transition the truck makes from being empty to being loaded with one full 45ft dry container. Arcs $(0, 1)$ and $(1, 9)$ allow the truck to start and finish its route, respectively.

In general case, let $\Theta_f = \{0, 1, \ldots, m_f - 1, m_f\}$ be the set of possible states for a type f truck. States $1, \ldots, m_f - 1$ represent the load configurations the truck can have along its route, with states 0 and m_f indicating the truck's start and final states, respectively. Let $\Sigma_f \subset (\Theta_f \times \mathcal{R})$ be the subset of ordered pairs $(h, r) \in (\Theta_f \times \mathcal{R})$ such that the service described by r can be undertaken by a type f truck in state h. The state transitions for a type f truck are then determined by a transition function $q_f : \Sigma_f \to \Theta_f; (h, r) \mapsto q_f(h, r)$. Therefore, the state transition graph for a type f truck is composed of nodes representing truck states $h \in \Theta_f$ and arcs representing transitions (h, h') such that $q_f(h, r) = h'$, with $(h, r) \in \Sigma_f$ and $h' \in \Theta_f$.

Referring back to the example of Fig. 2, let us denote heavy-duty trucks by $f = 1$, then we can define the set of truck states as $\Theta_1 = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$. As previously mentioned, states $1, \ldots, 8$ represent the load configurations a truck of this type can have, and states 0 and 9 are the truck's start and final states, respectively. Considering the services that a truck of this type can perform, we define the transition function q_1 as follows:

$$
\begin{aligned}
&q_1(0, \mathtt{start}) = 1, \quad q_1(1, (+, \mathtt{e20}')) = 2, \quad q_1(2, (+, \mathtt{e20}')) = 3, \quad q_1(3, (-, \mathtt{e20}')) = 2,\\
&q_1(2, (-, \mathtt{e20}')) = 1, \quad q_1(1, (+, \mathtt{f20}')) = 4, \quad q_1(4, (-, \mathtt{f20}')) = 1, \quad q_1(1, (+, \mathtt{e40}')) = 5,\\
&q_1(5, (-, \mathtt{e40}')) = 1, \quad q_1(1, (+, \mathtt{f40}')) = 6, \quad q_1(6, (-, \mathtt{f40}')) = 1, \quad q_1(1, (+, \mathtt{e45}')) = 7,\\
&q_1(7, (-, \mathtt{e45}')) = 1, \quad q_1(1, (+, \mathtt{f45}')) = 8, \quad q_1(8, (-, \mathtt{f45}')) = 1, \qquad q_1(1, \mathtt{end}) = 9.
\end{aligned}
$$

To model other truck types, we proceed as in the above example, as shown in Fig. 3. Fig. 3.1 shows the state transition graph of medium-duty trucks that can only haul empty 20ft and 40ft dry containers. Fig. 3.2 shows the state transition graph of heavy-duty trucks that haul 20ft dry and tank containers.

The proper definition of each truck's transition graph implicitly models capacity and load configuration constraints. Capacity constraints are enforced as there cannot be states and transitions allowing a truck to carry more containers when no load capacity remains. Moreover, there cannot be states and transitions allowing an empty truck to perform deliveries. Compatibility constraints between containers and trucks are enforced as a truck can only handle a container if this is allowed on the truck's transition graph. To illustrate this point, consider the trucks shown in Fig. 3. Note that only the medium-duty trucks can move empty 40ft dry containers, and only the heavy-duty trucks can move 20ft tank containers. We can also define states and transitions that enforce compatibility constraints between containers depending on their loads. To illustrate this point, refer to Fig. 2 and observe that full 20ft containers cannot be moved with other 20ft containers. However, it is possible to move two empty 20ft containers at a time.

Note that the state transition graphs contain no information on the geographic location where containers are loaded on a truck nor the time the loading occurs. The transition logic is used exclusively to model capacity and compatibility constraints. In Sections 4.2

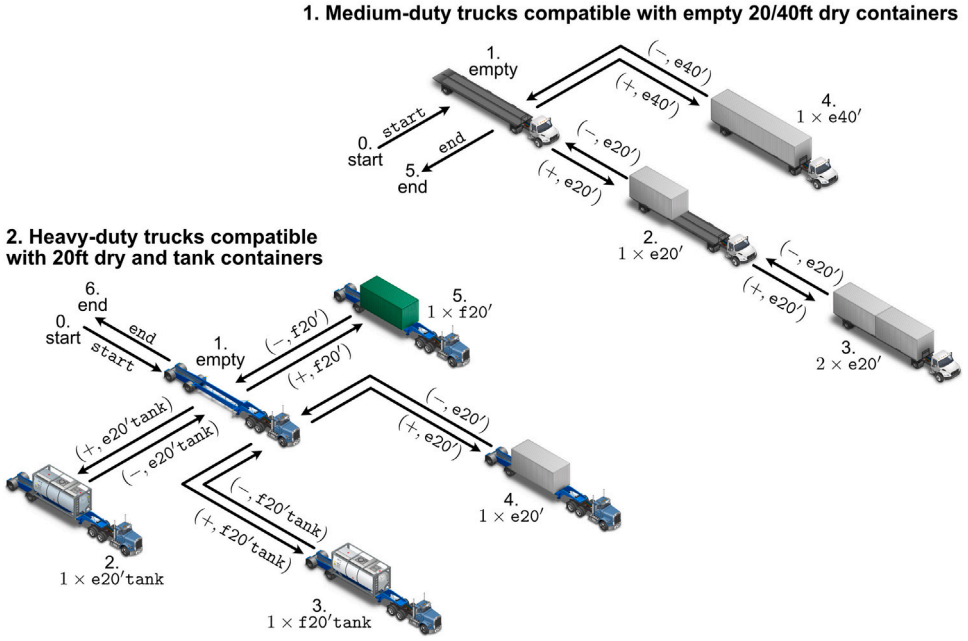


Fig. 3. Additional examples of state transition graphs.

and 4.3, we tackle the routing component of the DRP-HCL, connecting the trucks' state transitions with their movements across the graph of requests $\mathcal{G}$.

4.2. Three-index MIP formulation

This section describes a model for the DRP-HCL that uses classical routing traversal variables with three indices to indicate if a truck traverses an arc. Formally, let $\mathcal{T} = \{1, \ldots, T\}$ be the set of trucks. Also, let $f : \mathcal{T} \to \mathcal{F}; t \mapsto f(t)$ be a function that maps each truck to its corresponding type. An upper bound on the number of trucks in $\mathcal{T}$ can be obtained based on the number of requests compatible with each truck type $f \in \mathcal{F}$.

For each arc $(i, j) \in \mathcal{A}$ and truck $t \in \mathcal{T}$, let $x^t_{i,j}$ be a binary variable equal to one if and only if truck t travels arc (i, j), and zero otherwise. Furthermore, for each node $i \in \mathcal{N}$ and truck $t \in \mathcal{T}$, let B^t_i be the time at which truck t visits node i.

To associate the requests of a graph $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ (as the one shown in Fig. 1) with state transition graphs of trucks (as the ones shown in Figs. 2 and 3), we define the following sets:

- $\Delta^f_i = \{h' \in \Theta_f : q_f(h, r) = h',\ (h, r) \in \Sigma_f,\ \text{and } r = r_i\}$, which contains *the possible states in which a truck of type $f \in \mathcal{F}$ can be after visiting node $i \in \mathcal{N}$*; and
- $\Omega^f_{i,j} = \{(h, h') \in \Theta^2_f : q_f(h, r) = h',\ (h, r) \in \Sigma_f,\ h \in \Delta^f_i,\ \text{and } r = r_j\}$, which contains *the transitions (state at node i, state at node j) that a truck of type $f \in \mathcal{F}$ can perform in traversing an arc $(i, j) \in \mathcal{A}$.*

For each node $i \in \mathcal{N}$, truck $t \in \mathcal{T}$, and $h \in \Delta^{f(t)}_i$, let $y^{t,h}_i$ be a binary variable equal to one if and only if, after visiting node i, truck t is in state h, and zero otherwise. Additionally, for each arc $(i, j) \in \mathcal{A}$, truck $t \in \mathcal{T}$, and state transition $(h, h') \in \Omega^{f(t)}_{i,j}, |\Omega^{f(t)}_{i,j}| > 1$, let $z^{t,h,h'}_{i,j}$ be a binary variable equal to one if and only if truck t travels arc (i, j) making state transition (h, h'), and zero otherwise.

Using the definitions above, we formulate the DRP-HCL as the MIP of Eqs. (1) to (20).

$$\text{minimize} \quad \sum_{t \in \mathcal{T}} E^{f(t)} \sum_{j \in \mathcal{P}} x^t_{0,j} + \sum_{t \in \mathcal{T}} \sum_{(i,j) \in \mathcal{A}} c^{f(t)}_{i,j} x^t_{i,j}, \tag{1}$$

$$\text{subject to:} \quad \sum_{t \in \mathcal{T}} \sum_{j \in \mathcal{N}} x^t_{i,j} = 1, \quad \forall i \in \mathcal{P}, \tag{2}$$

$$\sum_{j \in \mathcal{N}} x^t_{i,j} - \sum_{j \in \mathcal{N}} x^t_{n+i,j} = 0, \quad \forall i \in \mathcal{P}, t \in \mathcal{T}, \tag{3}$$

$$\sum_{j \in \mathcal{N}} x^t_{0,j} = 1, \quad \forall t \in \mathcal{T}, \tag{4}$$

$$\sum_{i\in\mathcal{N}} x^t_{i,2n+1} = 1, \quad \forall t \in \mathcal{T}, \tag{5}$$

$$\sum_{j\in\mathcal{N}} x^t_{j,i} - \sum_{j\in\mathcal{N}} x^t_{i,j} = 0, \quad \forall i \in \mathcal{P}\cup\mathcal{D}, t \in \mathcal{T}, \tag{6}$$

$$B^t_j \geq B^t_i + s_i + \tau_{i,j} - M_{i,j}(1 - x^t_{i,j}), \quad \forall (i,j) \in \mathcal{A}, t \in \mathcal{T}, \tag{7}$$

$$e_i \leq B^t_i \leq l_i, \quad \forall i \in \mathcal{N}, t \in \mathcal{T}, \tag{8}$$

$$B^t_{n+i} \geq B^t_i + s_i + \tau_{i,n+i}, \quad \forall i \in \mathcal{P}, t \in \mathcal{T}, \tag{9}$$

$$2x^t_{i,j} \leq y^{t,h}_i + y^{t,h'}_j + 2(1 - z^{t,h,h'}_{i,j}), \quad \forall (i,j) \in \mathcal{A}, t \in \mathcal{T}, (h,h') \in \Omega^{f(t)}_{i,j}, \left|\Omega^{f(t)}_{i,j}\right| > 1, \tag{10}$$

$$\sum_{(h,h')\in\Omega^{f(t)}_{i,j}} z^{t,h,h'}_{i,j} = x^t_{i,j}, \quad \forall (i,j) \in \mathcal{A}, t \in \mathcal{T}, \left|\Omega^{f(t)}_{i,j}\right| > 1, \tag{11}$$

$$2x^t_{i,j} \leq y^{t,h}_i + y^{t,h'}_j, \quad \forall (i,j) \in \mathcal{A}, t \in \mathcal{T}, (h,h') \in \Omega^{f(t)}_{i,j}, \left|\Omega^{f(t)}_{i,j}\right| = 1, \tag{12}$$

$$x^t_{i,j} = 0, \quad \forall (i,j) \in \mathcal{A}, t \in \mathcal{T}, \left|\Omega^{f(t)}_{i,j}\right| = 0, \tag{13}$$

$$y^{t,h}_0 = 1, \quad \forall t \in \mathcal{T}, h \in \Delta^{f(t)}_0, \tag{14}$$

$$y^{t,h}_{2n+1} = 1, \quad \forall t \in \mathcal{T}, h \in \Delta^{f(t)}_{2n+1}, \tag{15}$$

$$\sum_{h\in\Delta^{f(t)}_i} y^{t,h}_i = \sum_{j\in\mathcal{N}} x^t_{i,j}, \quad \forall i \in \mathcal{P}\cup\mathcal{D}, t \in \mathcal{T}, \tag{16}$$

$$x^t_{i,j} \in \{0,1\}, \quad \forall (i,j) \in \mathcal{A}, t \in \mathcal{T}, \tag{17}$$

$$y^{t,h}_i \in \{0,1\}, \quad \forall i \in \mathcal{N}, t \in \mathcal{T}, h \in \Delta^{f(t)}_i, \tag{18}$$

$$z^{t,h,h'}_{i,j} \in \{0,1\}, \quad \forall (i,j) \in \mathcal{A}, t \in \mathcal{T}, (h,h') \in \Omega^{f(t)}_{i,j}, \left|\Omega^{f(t)}_{i,j}\right| > 1, \tag{19}$$

$$B^t_i \geq 0, \quad \forall i \in \mathcal{N}, t \in \mathcal{T}. \tag{20}$$

The objective function (1) minimizes the total fixed and routing costs. Constraints (2) and (3) ensure that each request is served exactly once and, for each request, the pickup and delivery nodes are visited by the same truck. Constraints (4) and (5) ensure, respectively, that each truck leaves and returns to the depot node exactly once. Note that the trucks that go straight from the start depot to the end depot incur no cost. Constraints (6) guarantee that the truck that arrives at a customer node also departs from it. Time-window constraints are written as (7) and (8); $M_{i,j}$ are big M values that can be set to $\max\{l_i + s_i + \tau_{i,j} - e_j, 0\}$, $\forall (i,j) \in \mathcal{A}$ (see Desaulniers et al., 2014). Constraints (9) are precedence constraints. They ensure that, for each request, the pickup node is visited before the delivery node.

Constraints (10) to (16) are core for this formulation. They model the state transitions made by the trucks along their routes, thus implicitly enforcing capacity and compatibility constraints. Constraints (10) and (11) set the state transition a truck t makes in traversing an arc (i,j) when more than one state transition is possible, i.e., when $|\Omega^{f(t)}_{i,j}| > 1$. Specifically, constraints (11) enforce that one transition variable $z^{t,h,h'}_{i,j}$, with $(h,h') \in \Omega^{f(t)}_{i,j}$, is set to one when truck t traverses arc (i,j). This transition variable, in turn, activates one of the constraints (10), enforcing the selection of the corresponding truck states h and h' at the tail and head nodes of arc (i,j), respectively. Constraints (12) perform the function of constraints (10) and (11) when only one state transition is possible, i.e., when $|\Omega^{f(t)}_{i,j}| = 1$. Constraints (13) discard the arcs that a truck cannot traverse due to the absence of valid state transitions, i.e., when $|\Omega^{f(t)}_{i,j}| = 0$. Constraints (14) to (16) ensure that a truck is in a unique state after visiting each node along its route; note that $|\Delta^{f(t)}_0| = |\Delta^{f(t)}_{2n+1}| = 1$.

Finally, constraints (17) to (20) set the domains of the variables. It should be noted that we perform the following trivial variable fixations: $x^t_{i,i} = 0,\ \forall i \in \mathcal{N}, t \in \mathcal{T}$ and $x^t_{n+i,i} = 0,\ \forall i \in \mathcal{P}, t \in \mathcal{T}$. Note also that constraints (13) discard the following arcs, for which there are no feasible state transitions: $x^t_{i,0} = 0,\ \forall i \in \mathcal{N}, t \in \mathcal{T}$; $x^t_{2n+1,i} = 0,\ \forall i \in \mathcal{N}, t \in \mathcal{T}$; $x^t_{i,2n+1} = 0,\ \forall i \in \mathcal{P}, t \in \mathcal{T}$; and $x^t_{0,j} = 0,\ \forall j \in \mathcal{D}, t \in \mathcal{T}$.

4.3. Fragment-based formulation and branch-and-cut algorithm

This section describes an approach to the DRP-HCL based on the idea of generating solution fragments to create complete solutions. In the realm of the vehicle routing problems (VRPs), this idea has been explored by authors such as Ray et al. (2014) and Soeanu et al. (2020), who proposed heuristic methods to solve multi-depot VRPs, and Alyasiry et al. (2019), who proposed a branch-and-cut algorithm that achieved state-of-the-art performance in solving the *pickup and delivery problem with time windows and last-in-first-out loading constraints* (PDPTWL).

We propose a branch-and-cut algorithm based on the algorithm of Alyasiry et al. (2019), which consists of creating solutions by combining route fragments, i.e., feasible sequences of pickups and deliveries that start and end with an empty vehicle. The algorithm involves two main steps:

1. **Create a space–time network.** This step starts by generating all fragments satisfying the problem constraints. Then, the time windows of pickup and delivery nodes are discretized into fixed intervals, which allows the creation of the so-called timed nodes. The latter ones, in turn, allow the definition of timed fragments (i.e., timed versions of fragments) and timed arcs that connect them. This step ends with the creation of a space–time network composed of a depot node, timed pickup and delivery nodes, timed arcs, and timed fragments.
2. **Define a space–time network flow formulation.** This step defines a space–time network flow formulation that chains timed fragments and timed arcs to form routes. The formulation involves the use of separation procedures that dynamically add subtour elimination constraints and time-window constraints.

Our algorithm to solve the DRP-HCL differs from the algorithm of Alyasiry et al. (2019) in two major aspects: the fragment generation process and the number of space–time networks that must be considered. Regarding the former, we propose a fragment generation procedure specific to the DRP-HCL, which uses the state transition logic proposed in Section 4.1. Regarding the latter, the space–time network flow formulation of the DRP-HCL usually involves multiple networks, one for each truck type. By contrast, the space–time network flow formulation of the PDPTWL involves a single network, as the PDPTWL assumes a homogeneous fleet.

We organize the description of the proposed method as follows: Section 4.3.1 describes the creation of the space–time networks associated with each truck type. Then, Section 4.3.2 defines the space–time network flow formulation of the DRP-HCL. Finally, Section 4.3.3 describes the overall structure of the solution algorithm.

4.3.1. Creation of space–time networks

We need to create a space–time network for each truck type involved in the DRP-HCL. With this aim, we consider the problem definition presented in Section 3 with a slight modification: Assume a graph $\mathcal{G} = (\mathcal{N}, \mathcal{A})$, with node set $\mathcal{N} = \{0, 1, \ldots, 2n\}$ and arc set $\mathcal{A} = \{(i, j) : i, j \in \mathcal{N}, i \neq j\}$. Now, trucks depart from and arrive at a unique depot node, denoted by 0. Because each truck type $f \in \mathcal{F}$ has a specific load capacity and can visit specific subsets of pickup and delivery nodes (depending on its compatible containers), it is necessary to define F independent space–time networks, one for each truck type f. Such networks are composed of the depot node, timed pickup and delivery nodes, timed arcs, and timed fragments.

Algorithm 1 describes the process that creates a space–time network for a truck type $f \in \mathcal{F}$. Function Create-Network takes as input graph $\mathcal{G}$, a truck type f, and a fixed time step δ. Line 2 creates graph $\mathcal{G}_f = (\mathcal{N}_f, \mathcal{A}_f)$, which includes the nodes and arcs a type f truck can visit depending on the containers with which it is compatible. Line 3 removes inadmissible arcs from graph $\mathcal{G}_f$, applying the rules described by Dumas et al. (1991) to eliminate arcs violating precedence, pairing, or time-window constraints. Line 4 finds $\mathcal{X}_f$, the fragment set of truck type f, as further detailed below. Line 5 creates the node set $\mathcal{N}'_f$, set of timed arcs $\mathcal{A}'_f$, and set of timed fragments $\mathcal{X}'_f$. This step uses the same time discretization procedure described by Alyasiry et al. (2019), with time step δ. Line 6 creates the space–time network $\mathcal{G}'_f = (\mathcal{N}'_f, \mathcal{A}'_f \cup \mathcal{X}'_f)$ for truck type f, which is returned from the routine on line 7.

Algorithm 1 Creation of space-time network for truck type f

```
1: function Create-Network(G, f, δ)
2:     G_f ← Get-Graph(G, f)
3:     Preprocessing(G_f)
4:     X_f ← Find-Fragments(f, G_f)
5:     N'_f, A'_f, X'_f ← TW-Discretization(G_f, X_f, δ)
6:     G'_f ← (N'_f, A'_f ∪ X'_f)
7:     return G'_f
```

Fragment generation. Algorithm 2 describes the operation of function Find-Fragments, which creates the fragment set $\mathcal{X}_f$ on line 4 of Algorithm 1. This function takes as input a truck type f and its associated graph $\mathcal{G}_f$. As a first step, set $\mathcal{X}_f$ is initialized on line 14. Then, on lines 15–27, a depth-first search approach (which uses a stack data structure) finds the fragments that comprise set $\mathcal{X}_f$. The fragments are found by enumerating the paths in $\mathcal{G}_f$ satisfying the problem constraints.

The path enumeration process consists of extending paths by appending nodes to them. In this case, a path comprises the following data:

- *stops*, the path's current sequence of visited stops;
- i, the path's current last visited node;
- h, the state of the truck after visiting node i; and
- st, the time at which service starts at node i.

Each path starts at the depot node with a type f truck in the empty state (line 16). The nodes eligible to be appended to a path are those reachable from the path's current last node (line 20). A node can only be appended to a path if the resulting path meets all the problem constraints, which is checked by function Extend-Path on line 22.

Algorithm 2 Creation of set of fragments $\mathcal{X}_f$

```
1:  function Extend-Path(path, j, f)
2:      h' ← Update-State(path.h, r_j, Θ_f)
3:      if h' == Nil then                          ▷ Check capacity and compatibility constraints.
4:          return Nil
5:      if j ∈ D and j − n not in path.stops then   ▷ Check precedence constraints.
6:          return Nil
7:      st' ← Calculate-Time(path.i, j, path.st)
8:      W ← Get-Pending-Deliveries(path.stops, j)
9:      if st' > l_j or not Check-TW(j, st', W) then   ▷ Check time-window constraints.
10:         return Nil
11:     stops' ← path.stops + [j]
12:     return Path{stops: stops', i: j, h: h', st: st'}
13: function Find-Fragments(f, G_f)
14:     X_f ← ∅
15:     let stack be a new stack data structure
16:     path ← Path{stops: [0], i: 0, h: f.empty, st: 0.0}
17:     stack.push(path)
18:     while not stack.is-empty() do
19:         path ← stack.pop()
20:         for each j ∈ {k : there is an arc (path.i, k) ∈ G_f} do
21:             if j not in path.stops then
22:                 path' ← Extend-Path(path, j, f)
23:                 if path' ≠ Nil then
24:                     if path'.h == f.empty then
25:                         X_f ← X_f ∪ {Create-Fragment(path')}
26:                     else
27:                         stack.push(path')
28:     return X_f
```

Function Extend-Path is defined on lines 1–12. The inputs of this function are a path, a candidate node to append j, and a truck type f. This function checks if the path can be extended to node j meeting capacity, compatibility, precedence, and time-window constraints, as described below.

Lines 2–4 check capacity and compatibility constraints. These constraints hold if a type f truck can make a valid state transition when traveling from the current last node of the path to node j. Specifically, if h is the state of the truck after visiting the current last node of the path and r_j is node j's service description, then the path can be extended to node j if there exists a state $h' \in \Theta_f$ such that $q_f(h, r_j) = h'$. Function Update-State on line 2 performs this checking, identifying state h' if it exists. Note how the state transition logic proposed in Section 4.1 allows us to easily check capacity and compatibility constraints in this single step.

Lines 5–6 check precedence constraints: a path can only be extended to a candidate delivery node if the truck has already visited the corresponding pickup node. Meanwhile, lines 7–10 check time-window constraints. Line 7 calculates st', the expected start of service time at node j given the path's current last visited node, i, and the time at which service started at node i, st. Line 8 finds set $\mathcal{W}$, which contains the requests that would remain pending to deliver should the path be extended to node j. Line 9 checks if service at node j can start within the node's time window and if the requests in set $\mathcal{W}$ can be delivered on time, given that the start of service time at node j is st'. The latter check is performed by function Check-TW.

If all checks are successful, lines 11 and 12 extend the path and return it: Node j is appended to the sequence of stops of the current path. Thus, node j becomes the new last visited node, with h' being the state of the truck after visiting node j and st' being the time at which service starts at node j. If one of the checks fails, the path cannot be extended, and function Extend-Path returns Nil.

Returning to the definition of function Find-Fragments, when a path is extended to a node j, line 24 checks if the truck is in the empty state after visiting that node, if so, function Create-Fragment creates a new fragment from the path. Each new fragment is included in set $\mathcal{X}_f$ (line 25). A fragment is created by removing the depot node from the start of a path and calculating the cost of traversing the fragment. For a fragment $g = (\phi_1, \ldots, \phi_M)$, such a cost is given by $\hat{c}_g^f = \sum_{k=1}^{M-1} c^f_{\phi_k, \phi_{k+1}}$. Once all fragments have been found, line 28 returns the resulting set $\mathcal{X}_f$.

Fig. 4 shows some examples of fragments, considering trucks that haul one full 20/40/45ft dry container or up to two empty 20ft dry containers. Note that there are two types of fragments that can be created depending on the trucks' load capacities. The first type of fragment corresponds to short pickup-to-delivery fragments, which result when the containers occupy the trucks' full load capacities, as shown in Fig. 4.1. The second type of fragment corresponds to longer fragments comprising consecutive pickups and deliveries, which result when trucks haul more than one container at once, as shown in Fig. 4.2. This example allows us to make

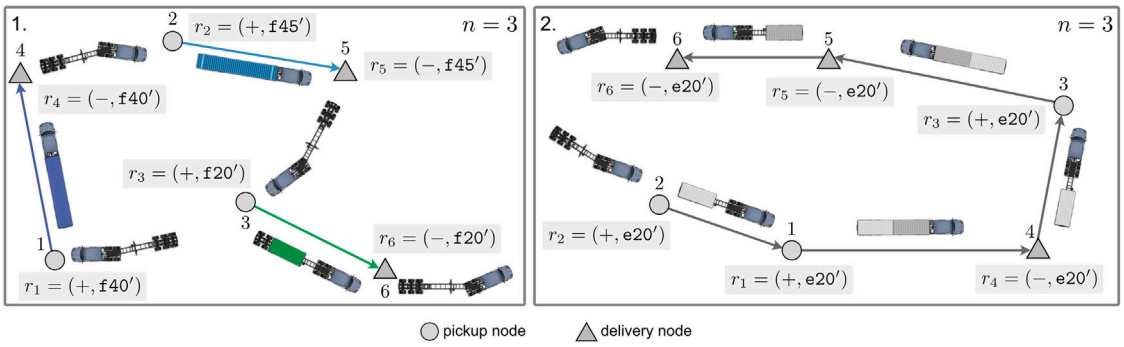


Fig. 4. Examples of fragments.

the following remark: The fragment generation process is trivial when containers occupy the trucks' full load capacities. However, when there are many containers that can be moved together, it may be necessary to explore many permutations of visits, a task that may require considerable time.

Time-window discretization. The time-window discretization procedure on line 5 of Algorithm 1 is detailed in Alyasiry et al. (2019); we briefly describe its outputs. For a truck type $f \in \mathcal{F}$, the node set is given by $\mathcal{N}'_f = \{0\} \cup \{\bigcup_{i \in \mathcal{P}_f \cup \mathcal{D}_f} \Lambda_i\}$, where $\mathcal{P}_f \cup \mathcal{D}_f$ is the set of pickup and delivery nodes a type f truck can visit, and Λ_i is the set of timed nodes derived from node i — a timed node represents node i at a given instant within its time window. Each timed fragment $w \in \mathcal{X}'_f$ is created from a fragment $g \in \mathcal{X}_f$. A timed fragment visits the same sequence of nodes and has the same traveling cost as the fragment from which it is derived. In addition, a timed fragment starts at a timed version of the first pickup node in its sequence of nodes and ends at a timed version of the last delivery node in its sequence of nodes. The set of timed arcs $\mathcal{A}'_f$ contains four types of arcs: (1) Timed empty arcs, which connect the end of a timed fragment (a timed delivery node) with the start of another one (a timed pickup node). (2) Timed waiting arcs, which connect the timed versions of the same node, moving forward in time. (3) Timed start arcs, which connect the depot to a timed pickup node. (4) Timed end arcs, which connect a timed delivery node to the depot. Let c_a^f be the cost a type f truck incurs in traveling arc $a \in \mathcal{A}'_f$. For timed start arcs, timed empty arcs, and timed end arcs, c_a^f is equal to the cost a type f truck incurs in traveling the corresponding original arcs. Furthermore, the cost of traveling timed start arcs includes the fixed cost of using the truck, E^f. Traveling timed waiting arcs does not incur any cost. The arrival times at the end nodes of timed fragments and timed empty arcs are rounded down to match the closest timed versions of such nodes. In this way, it is possible to join timed fragments and timed arcs at common discrete time points, thereby creating space–time paths. The time-window discretization procedure also applies dominance criteria to remove redundant timed fragments and timed empty arcs.

4.3.2. Space–time network flow formulation of the DRP-HCL

To find solutions to the DRP-HCL, we define space–time paths through the networks $\mathcal{G}'_f, f \in \mathcal{F}$. We must define the paths in such a way that they cover each request exactly once. Consider a truck type $f \in \mathcal{F}$. For any timed arc $a \in \mathcal{A}'_f$, we denote its start timed node (tail) as $a^- \in \mathcal{N}'_f$ and its end timed node (head) as $a^+ \in \mathcal{N}'_f$. Likewise, for any timed fragment $w \in \mathcal{X}'_f$, we denote its start timed node (tail) as $w^- \in \mathcal{N}'_f$ and its end timed node (head) as $w^+ \in \mathcal{N}'_f$. Let further $\mathcal{X}'_{f:p} = \{w \in \mathcal{X}'_f : w \text{ contains } p,\ p \in \mathcal{P}\}$ be the set of timed fragments of truck type f that visit node $p \in \mathcal{P}$.

For each truck type $f \in \mathcal{F}$ and timed arc $a \in \mathcal{A}'_f$, let λ_a^f be a binary variable equal to one if and only if timed arc $a \in \mathcal{A}'_f$ is selected, and zero otherwise. For each truck type $f \in \mathcal{F}$ and timed fragment $w \in \mathcal{X}'_f$, let μ_w^f be a binary variable equal to one if and only if timed fragment $w \in \mathcal{X}'_f$ is selected, and zero otherwise. Eqs.(21) to (27) define the space–time network flow formulation for the DRP-HCL.

$$\text{minimize} \quad \sum_{f \in \mathcal{F}} \sum_{a \in \mathcal{A}'_f} c_a^f \lambda_a^f + \sum_{f \in \mathcal{F}} \sum_{w \in \mathcal{X}'_f} \hat{c}_w^f \mu_w^f, \tag{21}$$

$$\text{subject to:} \quad \sum_{f \in \mathcal{F}} \sum_{w \in \mathcal{X}'_{f:p}} \mu_w^f = 1, \quad \forall p \in \mathcal{P}, \tag{22}$$

$$\sum_{a \in \mathcal{A}'_f, a^+ = i} \lambda_a^f + \sum_{w \in \mathcal{X}'_f, w^+ = i} \mu_w^f = \sum_{a \in \mathcal{A}'_f, a^- = i} \lambda_a^f + \sum_{w \in \mathcal{X}'_f, w^- = i} \mu_w^f, \quad \forall f \in \mathcal{F}, i \in \mathcal{N}'_f, \tag{23}$$

Subtour elimination constraints (24),

Time-window constraints (25),

$$\lambda_a^f \in \{0,1\}, \quad \forall f \in \mathcal{F}, a \in \mathcal{A}'_f, \tag{26}$$

$$\mu_w^f \in \{0,1\}, \quad \forall f \in \mathcal{F}, w \in \mathcal{X}'_f. \tag{27}$$

The objective function (21) minimizes the total fixed and routing costs. Constraints (22) check that each request is served exactly once. That is, from the timed fragments visiting a node $p \in P$, only one is selected. Constraints (23) are flow balance constraints. For the space–time network of each truck type, these constraints ensure that the number of timed arcs and timed fragments entering a timed node is equal to the number leaving it. Note that constraints (22) and (23) allow for subtours, i.e., paths not starting and ending at the depot. Subtours are eliminated with constraints (24), which are not defined explicitly in this model, but dynamically generated during solution time. Moreover, the rounding down of arrival times during the generation of the space–time networks can result in the violation of the upper limits of time-window constraints when multiple fragments are joined together, yielding the need for additional time-window constraints (25). As constraints (24), constraints (25) are also dynamically generated at solution time. The separation procedures that identify constraints (24) and (25) are described below. Finally, constraints (26) and (27) set the domains for the variables.

Separation problems. Consider a truck type $f \in \mathcal{F}$. Let $g(w) \in \mathcal{X}_f$ be the original fragment associated with timed fragment $w \in \mathcal{X}'_f$, and let $\mathcal{X}'_{f,g}$ be the set of all timed versions of a fragment $g \in \mathcal{X}_f$. Analogously, let $b(a) \in \mathcal{A}_f$ be the original delivery-to-pickup arc associated with timed empty arc $a \in \mathcal{A}'_f$, and let $\mathcal{A}'_{f,b}$ be the set of all timed versions of a delivery-to-pickup arc $b \in \mathcal{A}_f$.

Subtour elimination constraints. A subtour φ is a sequence of timed arcs and timed fragments that form a tour that does not include the depot. Let $\mathcal{E}_\varphi$ and $\mathcal{Z}_\varphi$ be, respectively, the sets of timed empty arcs and timed fragments in a subtour φ. Assume that $\xi \in \mathcal{F}$ is the type of the truck that traverses subtour φ.

Lemma 4.1. *The following constraint cuts subtour φ:*

$$\sum_{\alpha \in \mathcal{E}_\varphi} \sum_{a \in \mathcal{A}'_{\xi,b(\alpha)}} \lambda_a^\xi + \sum_{\sigma \in \mathcal{Z}_\varphi} \sum_{w \in \mathcal{X}'_{\xi,g(\sigma)}} \mu_w^\xi \leq |\mathcal{E}_\varphi| + |\mathcal{Z}_\varphi| - 1. \tag{24}$$

Proof. For all timed empty arcs $\alpha \in \mathcal{E}_\varphi$ and all timed fragments $\sigma \in \mathcal{Z}_\varphi$, subtour φ can be cut by imposing a Dantzig–Fulkerson–Johnson subtour elimination constraint limiting the number of selected timed empty arcs in $\mathcal{E}_\varphi$ and timed fragments in $\mathcal{Z}_\varphi$ to the cardinality of the sets $\mathcal{E}_\varphi$ and $\mathcal{Z}_\varphi$ minus one, that is:

$$\sum_{\alpha \in \mathcal{E}_\varphi} \lambda_\alpha^\xi + \sum_{\sigma \in \mathcal{Z}_\varphi} \mu_\sigma^\xi \leq |\mathcal{E}_\varphi| + |\mathcal{Z}_\varphi| - 1. \tag{24.1}$$

This constraint can be lifted by considering constraints (22), which impose that each pickup node $p \in \mathcal{P}$ is visited by a single timed fragment. In particular, this enforces the following:

$$\sum_{w \in \mathcal{X}'_{f,g}} \mu_w^f \leq 1, \qquad \forall g \in \mathcal{X}_f, f \in \mathcal{F}. \tag{24.2}$$

Moreover, since $\mu_\sigma^\xi = 1$ for any $\sigma \in \mathcal{Z}_\varphi$, we can write the following:

$$\sum_{w \in \mathcal{X}'_{\xi,g(\sigma)}} \mu_w^\xi \leq 1 = \mu_\sigma^\xi, \qquad \forall \sigma \in \mathcal{Z}_\varphi. \tag{24.3}$$

Similarly, no two timed versions of the same delivery-to-pickup arc can be used simultaneously due to constraints (22) and (23). Thus, we can write the following:

$$\sum_{a \in \mathcal{A}'_{\xi,b(\alpha)}} \lambda_a^\xi \leq 1 = \lambda_\alpha^\xi, \qquad \forall \alpha \in \mathcal{E}_\varphi. \tag{24.4}$$

Constraints (24.3) and (24.4) allow for the lifting of each variable μ_σ^ξ and λ_α^ξ in (24.1) to $\sum_{w \in \mathcal{X}'_{\xi,g(\sigma)}} \mu_w^\xi$ and $\sum_{a \in \mathcal{A}'_{\xi,b(\alpha)}} \lambda_a^\xi$, respectively. This yields the final version of the constraint presented in (24), thus completing the proof. □

Constraints (24) are implemented as lazy constraints. Whenever the solver finds a candidate integer solution, a callback procedure checks if all the paths that compose the solution are free from subtours. If not, the callback procedure adds the appropriate constraint (24) to remove this solution.

Time-window constraints. Let γ be a time-window infeasible path. Let $\mathcal{Z}_\gamma$ be the set of timed fragments belonging to the (time-window infeasible) shortest sequence of fragments in path γ, starting from the first two fragments onward. Let further $\mathcal{E}_\gamma$ be the set of timed empty arcs linking the timed fragments in $\mathcal{Z}_\gamma$. Assume that $\xi \in \mathcal{F}$ is the type of the truck that traverses path γ.

Lemma 4.2. *The following constraint cuts time-window infeasible path γ:*

$$\sum_{\alpha \in \mathcal{E}_\gamma} \sum_{a \in \mathcal{A}'_{\xi,b(\alpha)}} \lambda_a^\xi + \sum_{\sigma \in \mathcal{Z}_\gamma} \sum_{w \in \mathcal{X}'_{\xi,g(\sigma)}} \mu_w^\xi \leq |\mathcal{E}_\gamma| + |\mathcal{Z}_\gamma| - 1. \tag{25}$$

Table 2
Instance classification.

Characteristic	Classification	Description
Number of requests	25, 50, 75, 100, 200, 400, 600, 800, 1000	
Geographical distribution of locations	C1, C2, R, RC1, RC2	Section 5.2.1
Time-window lengths	W1, W2, W3	Section 5.2.2
Fleet of containers and trucks	F1, F2, F3, F4, F5	Section 5.2.3

Proof. The proof is analogous to the proof of the previous lemma. Considering the timed empty arcs and timed fragments that form the (time-window infeasible) shortest sequence of fragments in path γ, one imposes that their associated variables cannot simultaneously assume value one:

$$\sum_{\alpha \in \mathcal{E}_\gamma} \lambda_\alpha^\xi + \sum_{\sigma \in \mathcal{Z}_\gamma} \mu_\sigma^\xi \leq |\mathcal{E}_\gamma| + |\mathcal{Z}_\gamma| - 1. \tag{25.1}$$

Following the same argument as before, constraints similar to (24.3) and (24.4) can be used to lift constraint (25.1) to obtain constraint (25). □

Constraints (25) are also implemented as lazy constraints. Whenever an integer solution to model (21)–(24), (26)–(27) is found, all paths are verified for time-window feasibility, and any violation is removed with the addition of the associated constraint (25).

4.3.3. Branch-and-cut algorithm

Algorithm 3 outlines the branch-and-cut algorithm that solves the formulation based on fragments. We call this algorithm the *fragment-based approach*. Function Fragment-Based-Approach takes as input a graph of requests $\mathcal{G}$, a fleet of trucks $\mathcal{F}$, and a time step δ. First, lines 7–10 create the space–time networks associated with the given truck types, as described in Section 4.3.1. Then, function Solve-Formulation on line 11 uses a MIP solver to solve the formulation of Eqs. (21) to (27). The MIP solver uses lazy constraint callbacks, represented by function Callback-Check, to check if the solutions are feasible regarding the time-window constraints and do not have subtours. Finally, line 12 returns the solution found by the MIP solver.

Algorithm 3 Branch-and-cut algorithm

```
1:  function Callback-Check(solution)
2:      if solution contains subtours then
3:          add lazy constraints (24)
4:      if solution contains time-window infeasible paths then
5:          add lazy constraints (25)
6:  function Fragment-Based-Approach(G, F, δ)
7:      Networks ← ∅
8:      for f ∈ F do
9:          G'_f ← Create-Network(G, f, δ)
10:         Networks ← Networks ∪ {G'_f}
11:     solution ← Solve-Formulation(Networks, Callback-Check)
12:     return solution
```

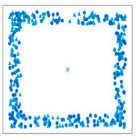
5. Computational results

5.1. Hardware and software

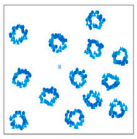
We implemented the three-index MIP formulation and the fragment-based approach using Python with the Gurobi 9.0 API. To analyze the results, we used Python (with the scikit-learn 1.0.2, xgboost 1.5.0, and shap 0.40.0 libraries) and R. We conducted the experiments on a server featuring an Intel Xeon E5-4650 CPU @ 2.70 GHz and 256 GB RAM.

5.2. Description of the problem instances

We developed a set of benchmark instances aiming to represent diverse and realistic drayage routing scenarios to test the proposed three-index MIP formulation and fragment-based approach. The instances are classified based on their time-window setting, fleet of containers and trucks, geographical distribution of locations, and number of service requests, as indicated in Table 2.



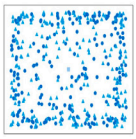
C1: The pickup and delivery locations of each request are located on opposite sides of the region. Thus, long distances are expected between the pickup and delivery locations of each request. The depot is located at the center of the region.



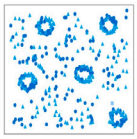
C2: The pickup and delivery locations of each request are clustered together, and there is a truck type compatible with all the requests of each cluster. Thus, the distances between pickup and delivery locations are expected to be short. The depot is randomly located.



R: All locations are randomly distributed (including the depot).



RC1: Locations are either randomly distributed or clustered as in C1 instances. The depot is located at the center of the region.



RC2: Locations are either randomly distributed or clustered as in C2 instances. The depot is randomly located.

Fig. 5. Geographical distributions of locations.

Table 3
Fleet classification based on compatibility between trucks and containers.

Fleet	Compatibility	Remarks
F1	Very high	All trucks can haul any container.
F2	Very high	All trucks can haul any container.
F3	High	Heavy-duty trucks can haul any container while medium-duty trucks can haul only empty containers.
F4	Medium	No truck is compatible with all containers: only empty 40ft containers are compatible with all three truck types, the other containers are compatible with only one or two truck types.
F5	Low	Each truck is compatible with a very small number of containers. Furthermore, each container is compatible with only one truck type.

5.2.1. Geographical distributions of locations

Fig. 5 describes the geographical distributions of locations: C1, C2, R, RC1, and RC2. Note that in instances with 200 or fewer requests, the locations lie in a square area of 30 mi × 30 mi. In instances with 400 or more requests, the locations lie in a square area of 60 mi × 60 mi. Furthermore, we assume that for any arc $(i, j) \in \mathcal{A}$, $\tau_{i,j}$ is equal to the Euclidean distance between i and j ($\tau_{i,j}$ is expressed in minutes).

5.2.2. Time windows

Let PH be the planning horizon length. For each request $i = 1, \ldots, n$, we generated the time windows of its pickup location, $[e_i, l_i]$, and delivery location, $[e_{n+i}, l_{n+i}]$, in such a way that e_i, l_i, e_{n+i}, and l_{n+i} are integers randomly selected from the range $[0, PH]$, $e_i \geq \tau_{0,i}$, $e_{n+i} \geq e_i + s_i + \tau_{i,n+i}$, $l_{n+i} \geq l_i + s_i + \tau_{i,n+i}$, and $l_{n+i} \leq PH - \tau_{n+i,2n+1} - s_{n+i}$. In line with the problem instances of Namboothiri and Erera (2008), we set $PH = 600$ min for all instances, and considered time-window lengths equal to 60 min and 120 min. We defined three types of instances according to the requests' time windows: instances W1 feature 60-min time windows, instances W2 feature 120-min time windows, and instances W3 feature 60- and 120-min time windows. In all instances, $s_i = s_{n+i} = 30$ min for $i = 1, \ldots, n$.

5.2.3. Fleets

Fig. 6 describes the considered fleets: F1, F2, F3, F4, and F5. Note that each fleet involves specific container and truck types. Moreover, each truck type is compatible with specific container types and has a specific load capacity (shown by arrows from truck to container). In defining the trucks' load capacities, we considered publicly available specification sheets of container chassis, such as those published online by Max-Atlas Equipment International Inc (2022).

Note that there are two characteristics of the fleets that can affect the difficulty of solving an instance: the compatibility between trucks and containers and the number of containers that can be hauled in a 2 × 20′ load configuration. Tables 3 and 4 show the classification of the fleets according to these two characteristics. We will refer to such classifications when analyzing the obtained results in the next sections.

The trucks' fixed and routing costs are given in US dollars and are shown at the bottom of Fig. 6. We calculated such costs considering data on costs of trucks from the studies of Vijayagopal and Rousseau (2021) and Giuliano et al. (2021), as described below.

We assume that all requests can be served in a single day. Therefore, the trucks' fixed costs correspond to daily costs. Specifically, for a truck type $f \in \mathcal{F}$, we assume its fixed cost E^f to be the daily cost of purchasing the truck with a 5% annual percentage rate (APR) loan, repaid over a five-year period. For medium-duty trucks, we considered the purchase price of Class 6 diesel trucks reported by Vijayagopal and Rousseau (2021). For heavy-duty trucks, we considered the purchase price of 2020 heavy-duty diesel trucks reported by Giuliano et al. (2021).

We assume that all trucks travel at 30 mph and that, for a truck type $f \in \mathcal{F}$, its routing costs $c^f_{i,j}, (i, j) \in \mathcal{A}$, encompass fuel, maintenance, and driver costs (calculated on a per-minute basis). To calculate the fuel costs, we considered the diesel cost per gallon diesel equivalent (DGE) reported by Giuliano et al. (2021) and fuel economy rates, given in miles per DGE (mi/DGE), of medium- and heavy-duty trucks. For medium-duty trucks, we considered the fuel economy rate of Class 6 diesel trucks reported

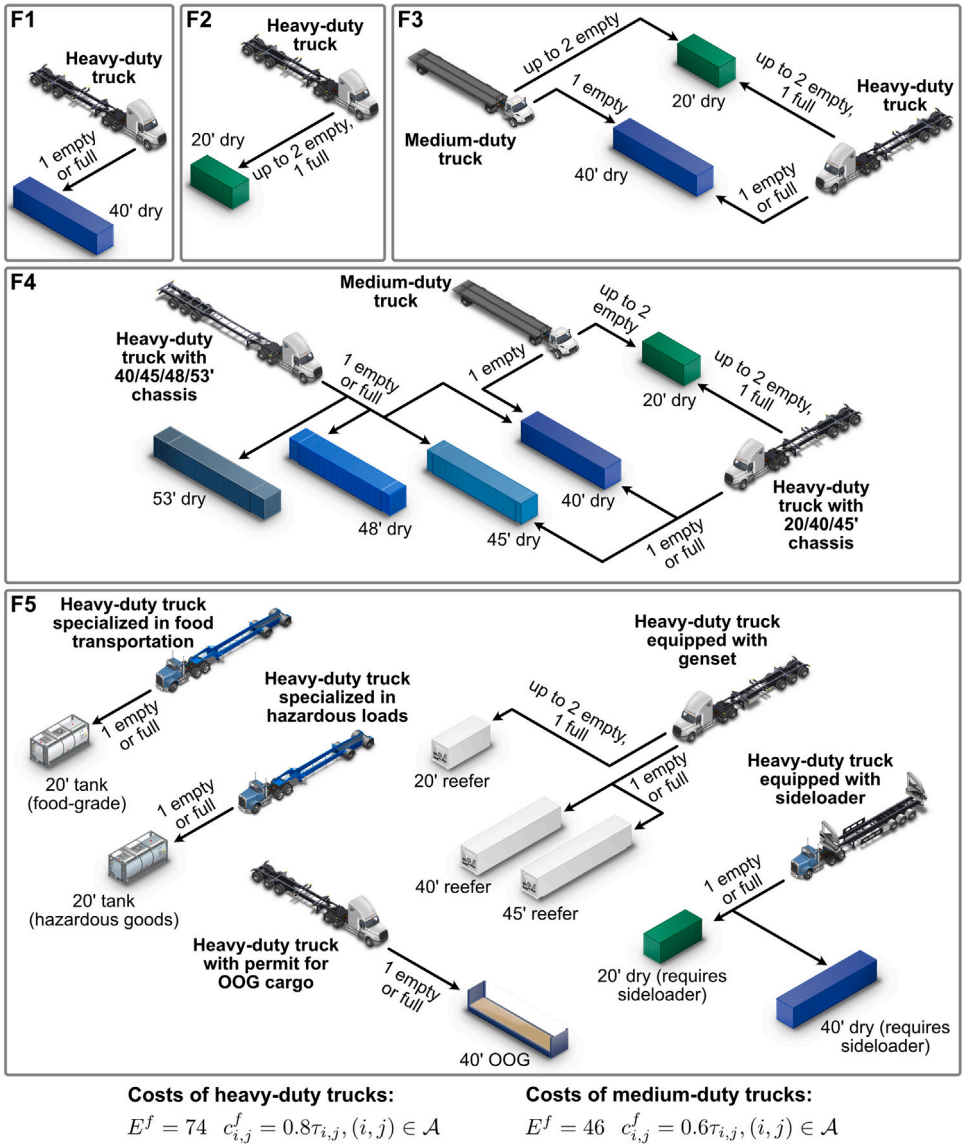


Fig. 6. Description of the fleets.

Table 4
Fleet classification based on the ratio of containers that can be moved in a 2 × 20′ load configuration.

Fleet	Ratio of 2 × 20′ loads	Remarks
F1	None	This fleet does not involve 20ft containers.
F2	Very high	There are two container types: full 20ft dry and empty 20ft dry. One of them, the empty 20ft dry, can be hauled in a 2 × 20′ load configuration.
F3	High	There are four container types: full 20ft dry, full 40ft dry, empty 20ft dry, and empty 40ft dry. One of them, the empty 20ft dry, can be hauled in a 2 × 20′ load configuration.
F4	Low	There are 10 container types (five container sizes, full and empty). From them, only one, the empty 20ft dry, can be hauled in a 2 × 20′ load configuration.
F5	Very low	There are 16 container types (eight container categories, full and empty). From them, only one, the empty 20ft reefer, can be hauled in a 2 × 20′ load configuration.

by Vijayagopal and Rousseau (2021). For heavy-duty trucks, we considered the fuel economy rate of 2020 heavy-duty diesel trucks reported by Giuliano et al. (2021). To calculate the maintenance costs for all trucks, we considered the per-mile maintenance costs

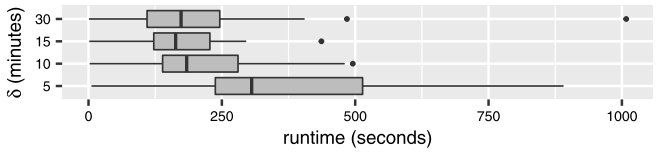


Fig. 7. Runtimes of the fragment-based approach using different values of δ.

of diesel trucks reported by Giuliano et al. (2021). Finally, we calculated the driver costs for all trucks assuming that drivers are paid 60 cents per mile (CPM).

5.3. Tuning the fragment-based approach

The runtime performance of the fragment-based approach can vary depending on the time step of the time-window discretization, δ. Thus, we conducted a set of experiments to compare the runtimes of solving a set of instances with different values of δ, aiming to find a value yielding shorter runtimes. After preliminary testing, we noted that 25-request instances could be solved quickly regardless of the value of δ used. On the other hand, we noted that instances with 100 or more requests can require very long times to solve. Thus, for this set of experiments, we considered only 50- and 75-requests instances, aiming to run the experiments faster. From all the 50- and 75-request instances we generated, we selected 30 instances and solved them considering values of δ equal to 5, 10, 15, and 30 min. Fig. 7 shows the obtained results.

We compared the (square-root transformed) runtimes using a one-way analysis of variance (ANOVA) test. This test indicated that the runtimes significantly differed depending on the value of δ used ($F_{(3,116)} = 8.59$, $p < 0.001$). Tukey's post-hoc tests indicated that the fragment-based approach had similar runtimes when setting δ to 10, 15, or 30 min (p-values > 0.78 in all comparisons). On the other hand, the runtimes obtained when setting δ to 5 min were significantly longer compared with the runtimes obtained when setting δ to 10, 15, or 30 min (p-values < 0.001 in all comparisons). Thus, we used the fragment-based approach setting δ to 15 min, as this value yielded the lowest median runtime.

5.4. Results for 25-, 50-, and 75-request instances

We used the three-index MIP formulation and fragment-based approach to solve instances with 25, 50, and 75 requests (225 instances in total). For each solver run, we set the maximum runtime to one hour (3600 s). We first solved all instances with the fragment-based approach, which allowed us to determine the number of trucks involved in the optimal solutions. Then, using such information, we limited the number of trucks available in the instances solved with the three-index MIP formulation, seeking to reduce the solving times. Table 5 summarizes the results of this set of experiments, grouping the instances by the number of requests, the geographical distribution of locations, the type of fleet, and the time-window setting. The tables enumerate the instances solved to proven optimality, the instances for which a feasible solution was found within the time limit, and the instances unsolved within the time limit. The average runtimes to find the proven optimal solutions are presented in seconds or shown as "N/A" if no proven optimal solutions were found. The results of the fragment-based approach and the three-index MIP formulation are displayed in the rows marked as "fragments" and "three-index", respectively. Moreover, the detailed solutions of this set of experiments are available in Tables A.9–A.11.

The fragment-based approach solved this set of instances to optimality in a short time. Meanwhile, the three-index MIP formulation was solved to proven optimality for 72 out of 75 instances with 25 requests, 48 out of 75 instances with 50 requests, and 37 out of 75 instances with 75 requests. Moreover, with this formulation, we found feasible solutions for all instances with 25 requests, while it was not possible to find any feasible solution for 20 of the instances with 50 requests and 32 of the instances with 75 requests.

Note that the results of the three-index MIP formulation were artificially improved by considering as input the number of trucks obtained by the fragment-based approach. In spite of this fact, the fragment-based approach still outperformed the three-index MIP formulation on all but one instance.

5.5. Results for larger instances

To further assess the performance of the fragment-based approach, we considered larger instances comprising 100, 200, 400, 600, 800, and 1000 requests (450 instances in total). In this set of experiments, we divided the solution process into two stages. The first stage consisted of conducting the fragment generation process with a time limit of one hour. The second stage consisted of solving the space–time network flow formulation and only involved the instances whose fragments could be generated within the specified time limit. In this case, we set the solver's maximum runtime to two hours (7200 s).

Tables 6 and 7 summarize the results of this set of experiments, grouping the instances by the number of requests, the geographical distribution of locations, the fleet, and the time-window setting. Table 6 enumerates the instances solved to proven optimality. The average solving times are presented in seconds or shown as "N/A" if no proven optimal solutions were found.

Table 5
Summary of results on 25-, 50-, and 75-request instances.

Class	Approach	Proven optimal			Avg. time (s)			Time limit (1 h)					
								Feasible solution			Unsolved		
		25	50	75	25	50	75	25	50	75	25	50	75
Geographical distribution of locations													
C1	Fragments	15	15	15	0.09	1.44	2.21	0	0	0	0	0	0
	Three-index	14	11	9	19.89	254.97	437.30	1	0	2	0	4	4
C2	Fragments	15	15	15	0.24	4.15	195.41	0	0	0	0	0	0
	Three-index	15	7	6	84.14	244.35	239.57	0	5	1	0	3	8
R	Fragments	15	15	15	0.15	4.35	8.77	0	0	0	0	0	0
	Three-index	15	11	8	102.32	149.11	380.33	0	0	1	0	4	6
RC1	Fragments	15	15	15	0.15	0.63	8.50	0	0	0	0	0	0
	Three-index	13	11	8	121.14	298.83	156.42	2	1	1	0	3	6
RC2	Fragments	15	15	15	0.19	1.95	42.27	0	0	0	0	0	0
	Three-index	15	8	6	58.76	403.80	407.38	0	1	1	0	6	8
Container and truck types													
F1	Fragments	15	15	15	0.20	0.92	3.35	0	0	0	0	0	0
	Three-index	15	13	11	18.46	260.11	219.30	0	2	0	0	0	4
F2	Fragments	15	15	15	0.22	7.72	151.51	0	0	0	0	0	0
	Three-index	13	0	0	371.81	N/A	N/A	2	0	0	0	15	15
F3	Fragments	15	15	15	0.25	1.56	62.29	0	0	0	0	0	0
	Three-index	14	7	1	29.19	668.62	961.96	1	4	3	0	4	11
F4	Fragments	15	15	15	0.09	1.36	39.18	0	0	0	0	0	0
	Three-index	15	13	10	0.57	333.69	649.83	0	1	3	0	1	2
F5	Fragments	15	15	15	0.06	0.96	0.83	0	0	0	0	0	0
	Three-index	15	15	15	0.26	18.21	149.25	0	0	0	0	0	0
Time windows													
W1	Fragments	25	25	25	0.06	0.35	0.76	0	0	0	0	0	0
	Three-index	25	19	16	15.37	54.41	104.72	0	1	2	0	5	7
W2	Fragments	25	25	25	0.33	6.09	144.30	0	0	0	0	0	0
	Three-index	23	12	8	148.52	370.99	902.96	2	4	1	0	9	16
W3	Fragments	25	25	25	0.10	1.07	9.24	0	0	0	0	0	0
	Three-index	24	17	13	72.14	422.77	247.09	1	2	3	0	6	9
All	Fragments	75	75	75	0.16	2.50	51.43	0	0	0	0	0	0
	Three-index	72	48	37	76.83	264.02	327.33	3	7	6	0	20	32

Table 6
Large instances solved to proven optimality in the 2nd stage.

Class	Proven optimal solutions						Avg. time (s)					
	100	200	400	600	800	1000	100	200	400	600	800	1000
Geographical distribution of locations												
C1	15	14	15	13	12	10	17.67	193.27	24.18	45.85	56.93	120.68
C2	14	8	6	4	3	1	257.36	259.46	801.18	1812.88	3781.16	612.64
R	15	12	12	10	8	7	264.83	78.25	88.42	434.75	238.17	671.92
RC1	15	14	14	13	10	8	23.94	289.32	39.25	71.74	217.48	144.95
RC2	14	11	10	6	5	3	62.14	658.73	1122.33	479.79	2393.40	1476.52
Container and truck types												
F1	15	15	14	12	11	9	8.89	426.55	368.37	413.12	1116.55	460.97
F2	13	6	7	2	1	0	404.95	814.61	105.39	71.93	124.09	N/A
F3	15	11	11	8	4	1	209.38	287.86	900.74	279.99	126.57	255.79
F4	15	12	12	11	9	7	33.21	171.30	153.16	658.97	846.50	202.36
F5	15	15	13	13	13	12	2.19	33.92	27.89	108.98	580.23	524.25
Time windows												
W1	25	24	24	21	17	13	3.68	62.26	218.17	462.87	471.92	364.66
W2	23	15	13	11	7	6	340.64	577.23	298.77	401.41	358.65	529.07
W3	25	20	20	14	14	10	45.71	343.21	444.18	133.62	1252.90	419.71
All	73	59	57	46	38	29	124.24	288.42	315.85	347.97	738.79	417.66

Table 7 enumerates the instances for which the solution process timed out in the first or second stage. The detailed solutions of this set of experiments are available in Tables A.12 and A.13.

Table 7
Large instances for which the solution process timed out.

Class	1st stage time limit (1 h)						2nd stage time limit (2 h)											
	Unsolved						Feasible solutions						Unsolved					
	100	200	400	600	800	1000	100	200	400	600	800	1000	100	200	400	600	800	1000
Geographical distribution of locations																		
C1	0	1	0	2	3	5	0	0	0	0	0	0	0	0	0	0	0	0
C2	0	5	5	8	9	9	1	2	4	2	2	4	0	0	0	1	1	1
R	0	2	3	5	5	7	0	1	0	0	1	1	0	0	0	0	1	0
RC1	0	1	1	2	5	7	0	0	0	0	0	0	0	0	0	0	0	0
RC2	0	3	3	6	8	8	1	1	2	3	2	4	0	0	0	0	0	0
Container and truck types																		
F1	0	0	0	0	0	0	0	0	1	3	4	6	0	0	0	0	0	0
F2	0	8	7	13	14	15	2	1	1	0	0	0	0	0	0	0	0	0
F3	0	3	4	7	11	14	0	1	0	0	0	0	0	0	0	0	0	0
F4	0	1	1	3	5	7	0	2	2	1	0	1	0	0	0	0	1	0
F5	0	0	0	0	0	0	0	0	2	1	1	2	0	0	0	1	1	1
Time windows																		
W1	0	0	0	4	8	10	0	1	1	0	0	2	0	0	0	0	0	0
W2	0	8	8	10	12	14	2	2	4	3	4	4	0	0	0	1	2	1
W3	0	4	4	9	10	12	0	1	1	2	1	3	0	0	0	0	0	0
All	0	12	12	23	30	36	2	4	6	5	5	9	0	0	0	1	2	1

The fragment-based approach found a feasible solution for all 75 instances with 100 requests and solved 73 of them to proven optimality. However, there were several unsolved instances among instances with 200 or more requests. As one may expect, the numbers of unsolved instances increased along with the number of requests. Overall, 337 out of 450 instances entered the second stage of the solution process, and a feasible solution was found for 333 of them (302 proven optimal solutions).

5.6. Performance analysis

The results shown in Sections 5.4 and 5.5 indicate that the solvability of the three-index MIP formulation and the performance of the fragment-based approach varied depending on the instances' characteristics. As one may expect, the number of requests had a major impact on the results. In general, instances involving fewer requests were easier to solve, while instances involving more requests were harder. Note, however, that certain geographical distributions of locations, time-window settings, and fleet types also rendered some instances harder or easier to solve.

To analyze the extent to which each instance feature impacted the obtained results, we used a machine learning (ML) approach consisting of two steps. First, we used a classifier to determine the likelihood of the instances to be solved to proven optimality (within the given time limits) depending on their features. Second, we used a visualization tool to study how the instances' features impacted the classifier's predictions.

In this case, we used XGBoost classifiers, originally proposed by Chen and Guestrin (2016). Generally speaking, XGBoost is an iterative approach based on ensembles of decision (or regression) trees — a decision (or regression) tree is a method that uses a flowchart-like structure to predict the value of a target variable based on a set of input variables. At each iteration, XGBoost creates a tree and assesses its predictions using an evaluation metric (e.g., logistic loss for classification or mean squared error for regression). In the next iteration, the algorithm tries to reduce the prediction errors of the previous tree. This process is repeated over a number of iterations, seeking to improve the accuracy of the obtained predictions.

We considered two XGBoost classifiers, one involving the results of the three-index MIP formulation (225 instances) and the other involving the results of the fragment-based approach (675 instances). For both classifiers, we considered the following input variables based on the instances' features:

- Number of requests (n.requests).
- Average container compatibility (comp). Considering all involved truck types, this feature is calculated by averaging their ratios of compatible containers to the total number of containers, see Table 3.
- Ratio of $2 \times 20'$ loads ($2 \times 20'$). This feature corresponds to the ratio of containers that can be moved in a $2 \times 20'$ load configuration to the total number of containers, see Table 4.
- Average travel time between the pickup and delivery locations of the requests (t.time).
- Average time-window length of the pickup and delivery locations (tw.length).

We considered a binary target variable, indicating if an instance can be solved to proven optimality within the given time limits (1) or not (0).

We implemented the classifiers using Python with the xgboost 1.5.0 library. In addition, we used the scikit-learn 1.0.2 library for performance assessment and hyper-parameter tuning. In this case, the evaluation metric we considered was the accuracy, i.e., the fraction of correctly classified samples. We measured the models' accuracies using stratified 5-fold nested cross-validation with the

Table 8
Models' hyper-parameters.

Hyper-parameter	Hyper-parameter space	Chosen value	
		Three-index	Fragments
No. estimators	{50, 100, 150, 200}	100	150
Max. depth	{3, 4, 5, … , 100}	65	57
Min. child weight	{0.0, 0.1, 0.2, … , 10.0}	4.7	3.6
Gamma	{0.0, 0.1, 0.2, … , 10.0}	0.6	1.1

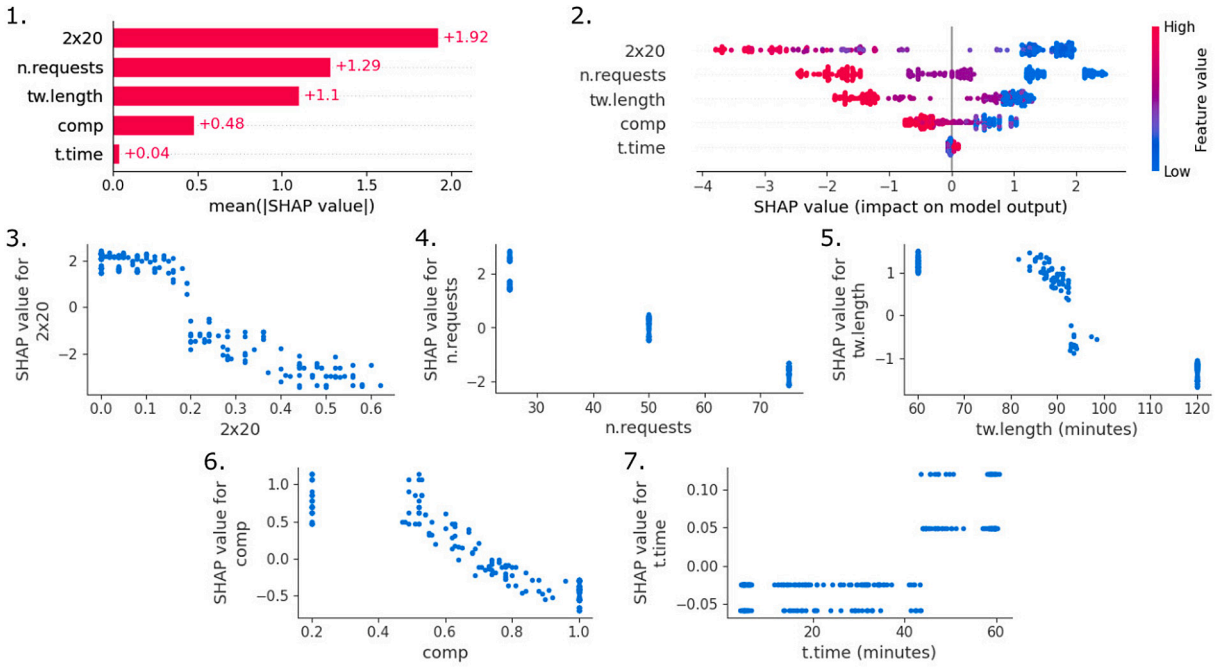


Fig. 8. SHAP plots of the XGBoost model used to classify the results of the three-index MIP formulation.

hyper-parameter spaces shown in Table 8. The model used to classify the results of the three-index MIP formulation yielded a mean accuracy of 0.92 (sd = 0.048). Meanwhile, the model used to classify the results of the fragment-based approach yielded a mean accuracy of 0.95 (sd = 0.021). To tune the hyper-parameters of both models, we used stratified 5-fold cross-validation; the chosen parameter values are presented in Table 8.

As visualization tool, we used SHapley Additive exPlanations (SHAP), an approach proposed by Lundberg and Lee (2017) and Lundberg et al. (2020) to explain the outputs of ML models. SHAP is based on a game-theoretic approach proposed by Shapley (1953) to assign fair payouts to players depending on their contributions to the total payout. In the realm of ML, SHAP allows quantifying the importance of the inputs to an ML model based on their contributions to a prediction.

We used Python with the shap 0.40.0 library to create SHAP visualizations of the XGBoost models. Then, based on such visualizations, we analyzed the influence of the instances' features on the solvability of the three-index MIP formulation, as discussed in Section 5.6.1, and on the performance of the fragment-based approach, as discussed in Section 5.6.2.

5.6.1. Performance of the three-index MIP formulation

Fig. 8 shows the SHAP plots of the model we used to classify the results of the three-index MIP formulation. Fig. 8.1 shows a global feature importance plot — the global importance of a feature is its mean absolute SHAP value. Fig. 8.2 shows a summary plot. In this plot, the instances' features are represented along the y-axis, and each point represents a single instance. For each feature, the position of a point along the x-axis represents the impact that such a feature had on the model's output for the instance represented by the point. Negative SHAP values indicate that an instance is less likely to be solved to proven optimality. Positive SHAP values indicate that an instance is more likely to be solved to proven optimality. As in Fig. 8.1, in Fig. 8.2 the features are positioned along the y-axis according to their importance, determined by their mean absolute SHAP values. Finally, Fig. 8.3 to .7 show dependence plots for each feature. In a dependence plot, each point represents an instance, the x-axis represents the values of a feature, and the y-axis represents the SHAP values for that feature. As in the summary plot, positive SHAP values indicate that an instance is likely to be solved to proven optimality and negative SHAP values indicate the opposite.

As it can be seen in Fig. 8.1 and .2, the feature that had the most impact on the solvability of the three-index MIP formulation was the ratio of $2 \times 20'$ loads, followed by the number of requests, the lengths of the time windows, and the compatibility between trucks and containers. The travel times had a negligible impact on the solvability of this formulation. Note that hard-to-solve instances generally involved many $2 \times 20'$ loads, many requests, long time windows, or high compatibility between trucks and containers. In contrast, easy-to-solve instances generally involved few $2 \times 20'$ loads, few requests, short time windows, or low compatibility between trucks and containers.

Fig. 8.3 to .7 indicate points at which the difficulty of solving the instances changed. As Fig. 8.3 indicates, instances likely to be solved to optimality have ratios of $2 \times 20'$ loads of less than 0.20 (e.g., F5 instances). The solution process was likely to time out for instances involving a ratio of $2 \times 20'$ loads greater than that (e.g., F2 instances). Fig. 8.4 shows that almost all 25-request instances were likely to be optimally solved, regardless of their other features. However, the outcome of the solution process for 50-request instances depended on the other features of these instances. For 75-request instances, the solution process was very likely to time out, regardless of the other features of these instances. Fig. 8.5 shows that W1 instances, which involve only short time windows, were likely to be optimally solved. So were W3 instances with low average time-window lengths. By contrast, W3 instances with higher average time-window lengths and W2 instances, which involve only long time windows, were unlikely to be optimally solved. Regarding the compatibility aspect of the instances, Fig. 8.6 shows that instances featuring compatibility ratios of less than 0.7 were likely to be optimally solved (especially F5 instances, which feature very low compatibility ratios). On the other hand, instances with compatibility ratios greater than 0.7 were unlikely to be optimally solved, especially some instances featuring full container compatibility (in particular F2 instances). Note that F1 instances also feature full container compatibility; however, such instances were more likely to be optimally solved than F2 instances, as their ratio of $2 \times 20'$ loads is zero, and such a feature had the most impact on the results. Fig. 8.7 indicates that instances likely to be optimally solved have average pickup-to-delivery travel times greater than 45 min. However, the impact of this feature on the predictions was negligible. So, we cannot conclude anything in this regard.

To explain the impact that the number of $2 \times 20'$ loads had on the solution process, we recall the formulation of Eqs. (1) to (20). Note that setting the state transitions for containers moved in a $2 \times 20'$ load configuration requires the use of constraints (10) and (11), which can lead to a hard-to-solve model. On the other hand, for containers that occupy the full load capacity of a truck, the state transitions are set by constraints (12), which reduces the difficulty of solving the model.

To explain the impact of container compatibility on the results, note that when there are trucks incompatible with some containers, constraints (13) cut down the number of nodes that such trucks can visit, which reduces the difficulty of solving the model. Therefore, instances featuring low container compatibility can be solved more easily.

Finally, regarding the impact that the time-window settings had on the obtained results, note that, in general, models involving long time windows have weaker linear relaxations than models involving short time windows, which explains why instances involving long time windows can be more difficult to solve.

5.6.2. Performance of the fragment-based approach

Fig. 9 shows the SHAP plots of the model we used to classify the results of the fragment-based approach. In this case, Fig. 9.1 and .2 show that the feature that had the largest impact on the performance of the fragment-based approach was the number of requests, followed by the ratio of $2 \times 20'$ loads, the travel times, and the length of the time windows. On the other hand, the container compatibility had a minor impact on the performance of this solution approach. In general, hard-to-solve instances involved many requests, many $2 \times 20'$ loads, short travel times, and long time windows. By contrast, easy-to-solve instances involved few requests, few $2 \times 20'$ loads, long travel times, and short time windows.

Fig. 9.3 to .7 indicate points at which the difficulty of solving the instances changed. Fig. 9.3 indicates that the fragment-based approach was likely to solve instances with 200 or fewer requests. Some 400-request instances were also likely to be optimally solved (depending on their other features). However, the solution process was prone to be interrupted for instances involving 600 or more requests. Fig. 9.4 indicates that instances with ratios of $2 \times 20'$ loads of less than 0.15 (e.g., F5 instances) were likely to be optimally solved. Instances with ratios of $2 \times 20'$ loads greater than that were shown to be more difficult, with the hardest instances being those with ratios of $2 \times 20'$ loads close to 0.5 (e.g., all F2 and some F3 instances). Fig. 9.5 shows that in instances likely to be optimally solved, the average pickup-to-delivery travel times are commonly greater than 40 min (e.g., all C1 and some RC1 instances). Instances with shorter average pickup-to-delivery travel times were unlikely to be optimally solved, especially instances where such average travel times are equal to 10 min or less (e.g., C2 instances). Fig. 9.6 indicates that W1 instances, which involve short time windows, were likely to be optimally solved. So were some W3 instances with low average time-window lengths. However, W3 instances with larger average time-window lengths and W2 instances, which involve only long time windows, were unlikely to be optimally solved. Fig. 9.7 shows that instances involving compatibility ratios of 0.55 or less were the most likely to be optimally solved. However, recall that this feature had a minor impact on the model's predictions.

To explain the impact that the number of $2 \times 20'$ loads had on the performance of the fragment-based approach, we should note that movements of containers in $2 \times 20'$ load configurations can be done in different permutations of visits, which can be associated with the creation of many fragments. In contrast, movements of containers that occupy the trucks' full load capacities require the generation of fewer (pickup-to-delivery) fragments (see Fig. 4). Naturally, the performance of the fragment-based approach depends largely on the number of fragments generated: instances that require the generation of large numbers of fragments are prone to be more difficult, while instances that require the generation of few fragments can be solved more easily.

Regarding the travel times between the requests' pickup and delivery locations, we should note that when many requests have pickup and delivery locations located nearby (which results in short travel times), there can be many valid routes that visit such

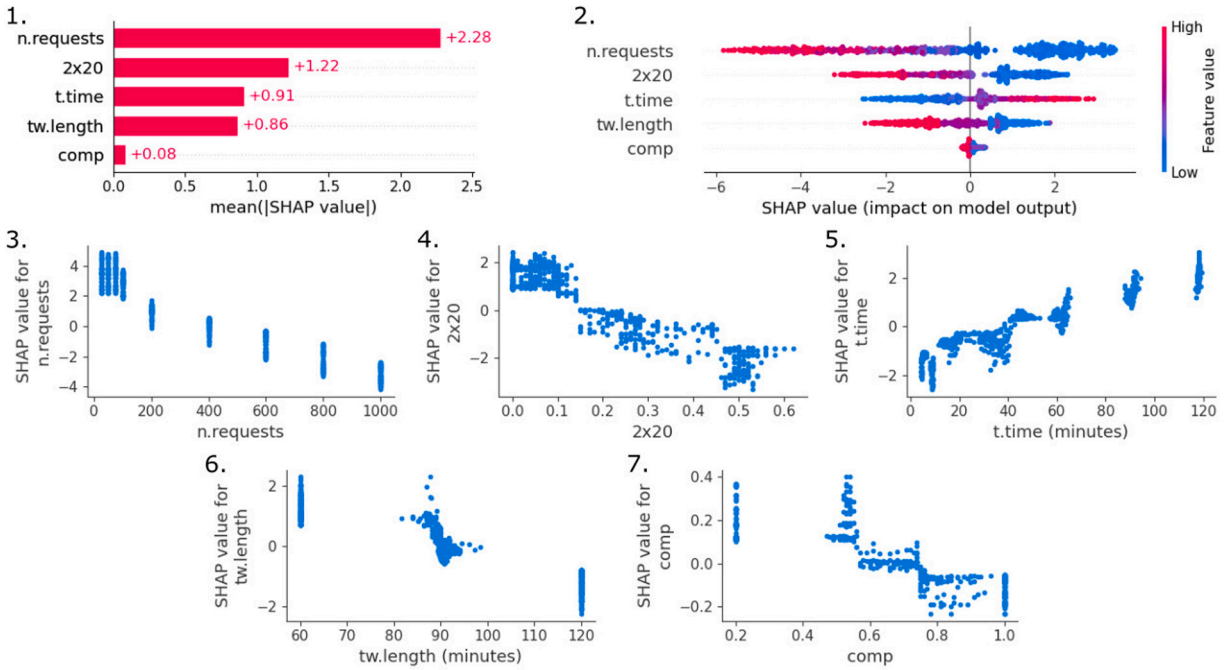


Fig. 9. SHAP plots of the XGBoost model used to classify the results of the fragment-based approach.

locations respecting the time-window constraints, which can hinder the solution process. On the other hand, when the pickup and delivery locations of many requests are located far apart from each other (which results in long travel times), there can be fewer valid routes to consider (due to the time-window constraints), which can ease the solution process.

Regarding the time-window settings, note that the time windows limit the routes that the trucks can follow. Instances involving stringent time windows (as W1 instances) can be easier to solve because, ultimately, the time windows reduce the number of feasible routes to consider.

Finally, regarding container compatibility, note that trucks incompatible with some containers are associated with smaller space–time networks (because the networks of such trucks do not include the nodes associated with incompatible containers). Therefore, instances featuring low compatibility between trucks and containers could eventually be solved more easily.

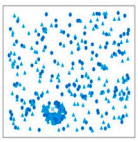
5.7. Using the fragment-based approach in fleet decision-making

In the previous section, we analyzed the performance of the fragment-based approach in solving DRP-HCL instances with different characteristics. Now, we assess the utility of the fragment-based approach in fleet decision-making. Specifically, we show how the fragment-based approach can help identify the best fleet compositions in three realistic scenarios of drayage routing. The first scenario is addressed in Section 5.7.1 and ponders the purchase of Class 4 and Class 6 trucks to move empty 20ft dry containers. The second scenario is addressed in Section 5.7.2 and ponders the purchase of diesel and natural gas (NG) hybrid heavy-duty trucks to haul 20ft, 40ft, and 45ft dry containers. Finally, the third scenario is addressed in Section 5.7.3 and analyzes the daily rental of container chassis to haul 20ft, 40ft, and 45ft dry containers.

All three scenarios involve two types of drayage movements: terminal and cross-town, described in Fig. 10. Terminal movements represent traditional drayage movements between terminal facilities and customer locations, possibly involving long-haul trips. Meanwhile, cross-town movements involve short drayage movements between rail ramps located nearby; these movements are described in detail by Pazour and Neubert (2013). In all cases, we assume a planning horizon that lasts 600 min and mixed time windows that last 60 or 120 min.

The computational experiments reported in this section involve three steps. First, for each of the three scenarios, we define six sub-scenarios characterized by a type of drayage movement and a number of requests, as follows: terminal movements and 25 requests, terminal movements and 50 requests, terminal movements and 75 requests, cross-town movements and 25 requests, cross-town movements and 50 requests, and cross-town movements and 75 requests. For each sub-scenario, we create 30 graphs of requests $\mathcal{G}$. Second, for each graph $\mathcal{G}$, we solve alternative DRP-HCL instances involving the same requests of graph $\mathcal{G}$ but different fleet compositions according to the scenario under consideration. Third, we identify the most suitable fleet compositions for each scenario, analyzing how the characteristics of the sub-scenarios affect the results.

Throughout this section, we assume that all costs are given in US dollars. In the scenarios discussed in Sections 5.7.1 and 5.7.2, the trucks' fixed costs correspond to the daily costs of purchasing the trucks with a 5% APR loan, repaid over a five-year period. In

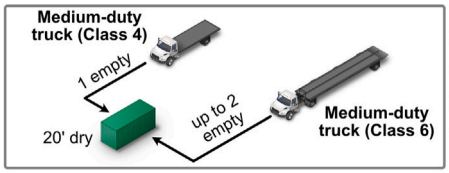


Terminal: Trucks move containers (full and empty) between facilities clustered within a terminal area and several customer locations, randomly distributed. There are requests for moving full containers between customer locations and terminal facilities (export requests) and vice versa (import requests). There are also requests for moving empty containers between customer locations (the so-called street turns) or in either direction between terminal facilities and customer locations. The terminal area is randomly located, and the depot is located within it.



Cross-town: Trucks move containers (full and empty) between rail ramps located nearby. Each rail ramp is represented as a cluster of pickup and delivery locations. The depot and rail ramps are randomly located.

Fig. 10. Types of drayage movements, based on the geographical distributions of locations.



Costs of medium-duty trucks (Class 4):
$E^f = 37 \quad c^f_{i,j} = 0.5\tau_{i,j}, (i,j) \in \mathcal{A}$

Costs of medium-duty trucks (Class 6):
$E^f = 46 \quad c^f_{i,j} = 0.6\tau_{i,j}, (i,j) \in \mathcal{A}$

Fig. 11. Class 4 vs. Class 6 trucks.

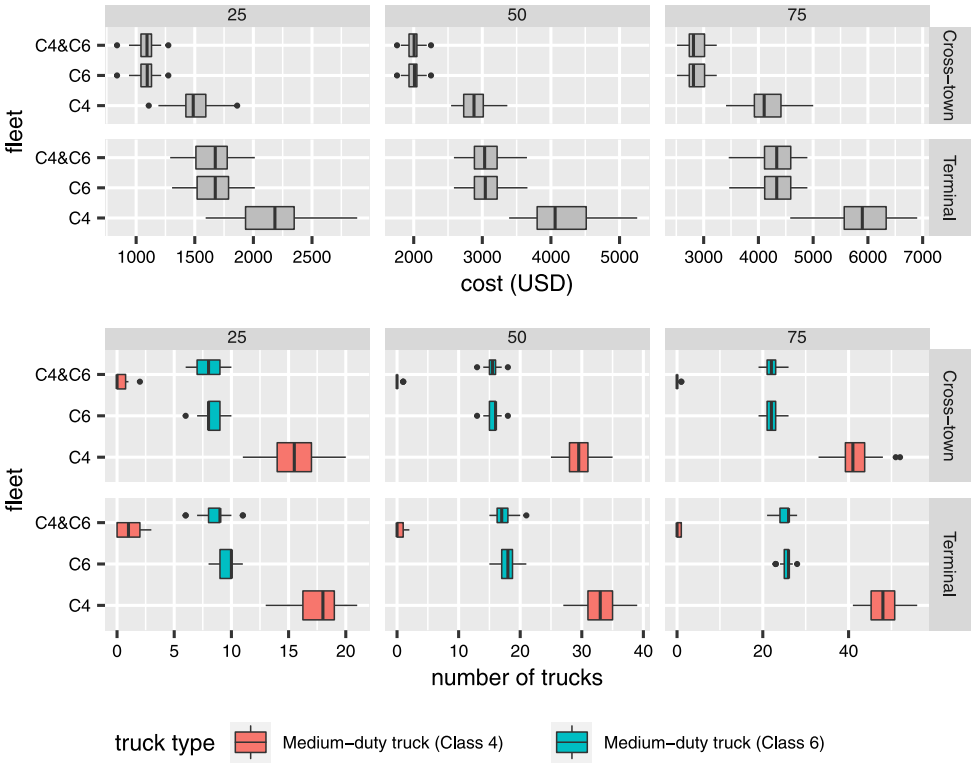
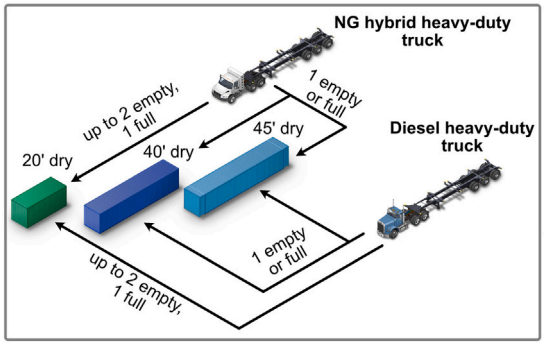


Fig. 12. Results: Class 4 vs. Class 6 trucks.

the scenario discussed in Section 5.7.3, the trucks' fixed costs correspond to the daily rental rates of container chassis. All scenarios assume that the trucks travel at 30 mph and that their routing costs involve fuel, maintenance, and driver costs (calculated on a per-minute basis). The fuel costs are calculated considering fuel costs per DGE and the trucks' fuel economy rates given in miles per DGE (mi/DGE). The maintenance costs are calculated based on per-mile maintenance costs. Finally, the driver costs are calculated assuming that drivers are paid 60 CPM.

5.7.1. Class 4 vs. Class 6 trucks

In this scenario, we consider the case of a drayage company that ponders the purchase of Class 4 and Class 6 trucks to move empty 20ft dry containers. Fig. 11 shows the costs, load capacities, and compatible containers of the trucks. Class 6 trucks can haul



Costs of diesel heavy-duty trucks:
$E^f = 74 \quad c^f_{i,j} = 0.8\tau_{i,j}, (i,j) \in \mathcal{A}$

Costs of NG hybrid heavy-duty trucks:
$E^f = 90 \quad c^f_{i,j} = 0.6\tau_{i,j}, (i,j) \in \mathcal{A}$

Fig. 13. Diesel vs. NG hybrid trucks.

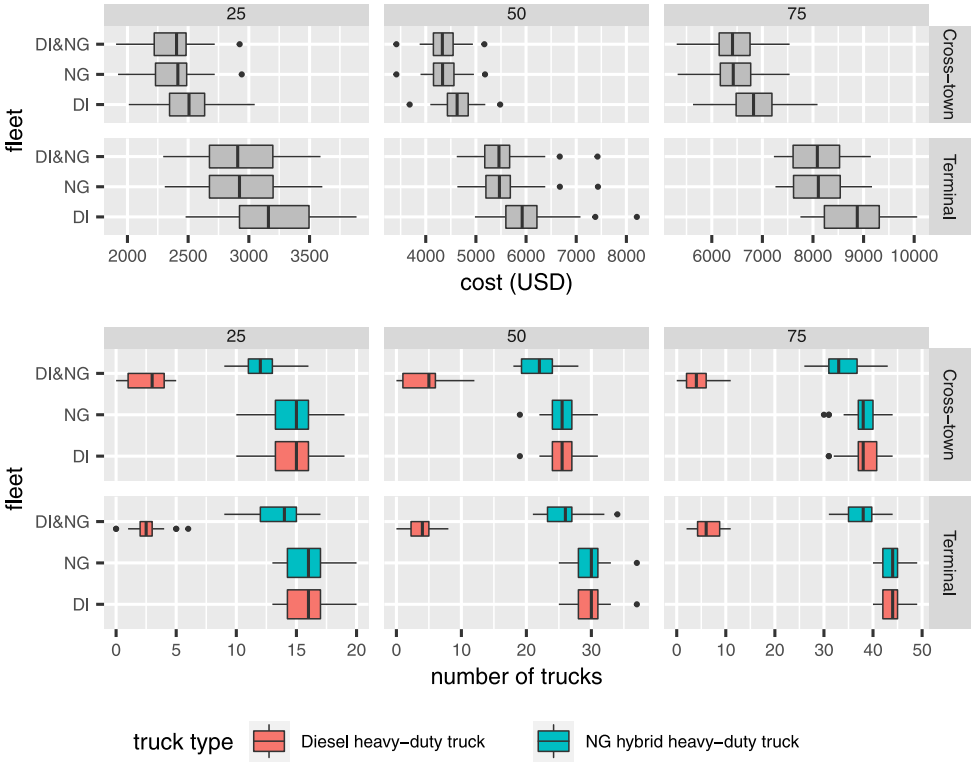


Fig. 14. Results: diesel vs. NG hybrid trucks.

up to two containers at a time, while Class 4 trucks can haul only one. However, Class 4 trucks have lower purchase prices and are also cheaper to operate.

To determine the fixed and routing costs of the trucks involved in this scenario, we considered the purchase prices and fuel economy rates (mi/DGE) of Class 4 and Class 6 diesel trucks reported by Vijayagopal and Rousseau (2021). In addition, we considered the fuel cost per DGE and per-mile maintenance costs of diesel trucks reported by Giuliano et al. (2021).

In this scenario, we compare the costs of hauling the containers using a homogeneous fleet of Class 4 trucks (C4), a homogeneous fleet of Class 6 trucks (C6), and a heterogeneous fleet composed of Class 4 and Class 6 trucks (C4&C6). Fig. 12 shows the results for all sub-scenarios.

A three-way ANOVA (with log-transformed costs) indicated that the main effects for type of drayage movement ($F_{(1,522)} = 2508.93$, $p < 0.001$), number of requests ($F_{(2,522)} = 5258.43$, $p < 0.001$), and fleet type ($F_{(2,522)} = 739.57$, $p < 0.001$) were all significant. The interaction between type of drayage movement and fleet type was also significant ($F_{(2,522)} = 8.42$, $p < 0.001$). In all sub-scenarios, Tukey's post-hoc tests found significant differences between the costs obtained with fleets C4 and C6 and C4 and C4&C6 (all p-values < 0.0001). In contrast, no differences were found between the costs obtained with fleets C4&C6 and C6 (all p-values > 0.94).

The results of the statistical tests allow us to make the following analysis: The acquisition of Class 6 trucks leads to significant cost savings, even though those trucks are more expensive to purchase and operate. This is an expected result, as Class 6 trucks

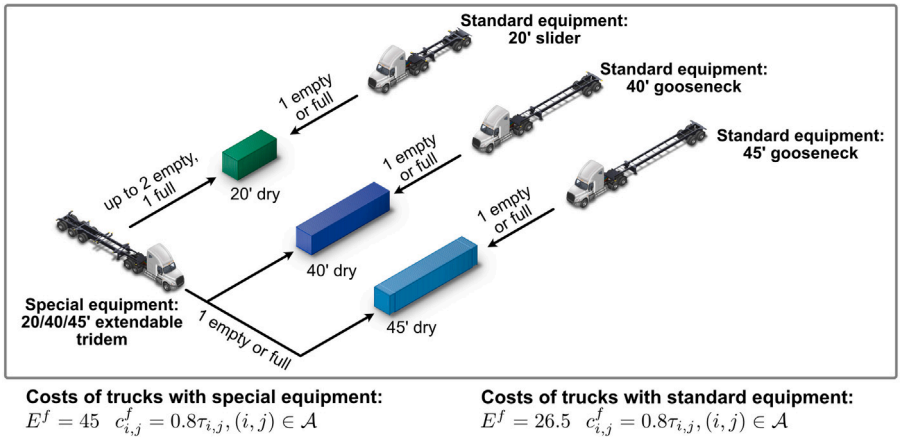


Fig. 15. Standard vs. special equipment.

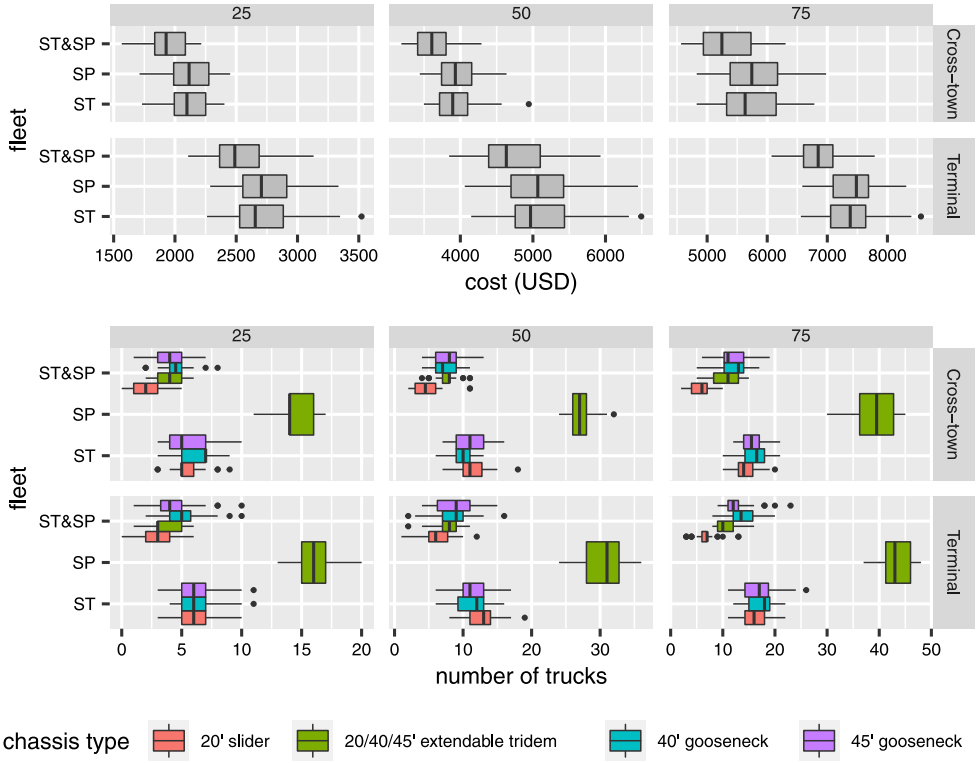


Fig. 16. Results: standard vs. special equipment.

enable more efficient routing due to their larger load capacity. Commonly, the most cost-effective fleets are composed entirely of Class 6 trucks. However, in a very small number of cases, the most cost-effective fleets are composed mostly of Class 6 trucks and some Class 4 trucks. In any case, the savings realized by fleets composed of Class 4 and Class 6 trucks are marginal when compared with the costs obtained by homogeneous fleets of Class 6 trucks. This observation is important because operating a homogeneous fleet can be more convenient regarding truck maintenance, as the variety of parts to maintain is limited. As one may expect, costs and fleet sizes increase when the routing involves more service requests and longer distances. However, the cost savings realized by Class 6 trucks are consistent regardless of the distances involved in the drayage movements and the number of service requests.

5.7.2. Diesel vs. NG hybrid trucks

In this scenario, we consider the case of a drayage company that ponders the purchase of heavy-duty trucks to move 20ft, 40ft, and 45ft dry containers. In this case, the aim is to compare the costs of trucks with different powertrains: diesel and NG hybrid.

Fig. 13 shows the costs, load capacities, and compatible containers of the trucks. Diesel trucks have a lower purchase price but are less fuel-efficient. In contrast, NG hybrid trucks have a higher purchase price but are more fuel-efficient. Both truck types have the same load capacity.

To determine the fixed and routing costs of the trucks involved in this scenario, we considered data from the study of Giuliano et al. (2021) related to 2020 purchase prices, fuel economy rates (mi/DGE), per-mile maintenance costs, and fuel costs per DGE of diesel and NG hybrid heavy-duty trucks. In this case, we assume that the company owns a fleet of chassis, which do not incur any cost.

In this scenario, we compare the costs of hauling the containers using a homogeneous fleet of diesel trucks (DI), a homogeneous fleet of NG hybrid trucks (NG), and a heterogeneous fleet composed of diesel and NG hybrid trucks (DI&NG). Fig. 14 shows the results for all sub-scenarios.

A three-way ANOVA (with log-transformed costs) indicated that the main effects for type of drayage movement ($F_{(1,522)} = 844.32$, $p < 0.001$), number of requests ($F_{(2,522)} = 5300.72$, $p < 0.001$), and fleet type ($F_{(2,522)} = 33.97$, $p < 0.001$) were all significant. Tukey's post-hoc tests found no significant differences between the costs obtained in the sub-scenario involving cross-town movements and 25 requests (all p-values > 0.07). For the sub-scenarios involving cross-town movements with 50 and 75 requests, Tukey's post-hoc tests found differences between the costs obtained with fleets DI and NG and DI and DI&NG (all p-values < 0.04). However, no significant differences were found between the costs obtained with fleets NG and DI&NG (p-values = 1.0). Likewise, in all sub-scenarios involving terminal movements, Tukey's post-hoc tests identified differences between the costs obtained with fleets DI and NG and DI and DI&NG (all p-values < 0.003), but no significant differences were found between the costs obtained with fleets NG and DI&NG (all p-values > 0.98).

Considering the results of the statistical tests, we can say that, in most cases, NG hybrid trucks can realize significant cost savings with respect to diesel trucks, indicating that the savings that NG hybrid trucks realize due to their better fuel efficiency outweigh their expensive purchase costs. Generally, the most cost-effective fleets are mainly composed of NG hybrid trucks and some diesel trucks, indicating that the cheaper purchase costs of diesel trucks outweigh their more expensive routing costs only if those trucks are used to haul few containers. However, the savings obtained by operating heterogeneous fleets of NG hybrid and diesel trucks are not significant when compared with the costs of operating homogeneous fleets of NG hybrid trucks. So, it may be more convenient to purchase only NG hybrid trucks, aiming to simplify fleet maintenance. Finally, we should also note that the savings realized by NG hybrid trucks may be less representative when the route planning involves few drayage movements and short distances.

5.7.3. Standard vs. special equipment

In this scenario, we consider the case of a drayage company that owns a fleet of heavy-duty semi-trucks and must rent a fleet of chassis to move 20ft, 40ft, and 45ft dry containers. Fig. 15 shows the costs, load capacities, and compatible containers of the trucks.

Considering the types of chassis offered by Flexi-Van Leasing LLC (2022), an American company specializing in chassis leasing services, we assume that there are four types of chassis available to rent: 20′ slider, 40′ gooseneck, 45′ gooseneck, and 20/40/45′ extendable tridem. The 20′ slider, 40′ gooseneck, and 45′ gooseneck are considered standard equipment. These chassis can carry one container at a time and are compatible with only one container type. Meanwhile, the 20/40/45′ extendable tridem is considered special equipment. This chassis is compatible with all three container types and can haul one (empty or full) 20/40/45ft container or two empty 20ft containers.

In this scenario, the fixed costs of the trucks correspond to daily rental rates of chassis. According to Flexi-Van Leasing LLC (2022), the daily rental rate per chassis is 26.5 USD for standard equipment and 45 USD for special equipment. The semi-trucks that pull the chassis do not incur any fixed cost because they are property of the drayage company. On the other hand, to estimate the semi-trucks' routing costs, we considered data from the study of Giuliano et al. (2021) related to the fuel cost per DGE, fuel economy rate (mi/DGE), and per-mile maintenance cost of heavy-duty diesel trucks. Considering that the daily rental rate of special equipment is higher than that of standard equipment, the aim is to identify the fleet of chassis that offers better value to the drayage company.

In this scenario, we compare the costs of hauling the containers using a heterogeneous fleet of 20′ slider, 40′ gooseneck, and 45′ gooseneck chassis (ST); a homogeneous fleet of 20/40/45′ extendable tridem chassis (SP); and a heterogeneous fleet of 20′ slider, 40′ gooseneck, 45′ gooseneck, and 20/40/45′ extendable tridem chassis (ST&SP). Fig. 16 shows the results for all sub-scenarios.

A three-way ANOVA (with log-transformed costs) indicated that the main effects for type of drayage movement ($F_{(1,522)} = 1035.58$, $p < 0.001$), number of requests ($F_{(2,522)} = 5367.32$, $p < 0.001$), and fleet type ($F_{(2,522)} = 45.21$, $p < 0.001$) were all significant. In all sub-scenarios, Tukey's post-hoc tests found significant differences between the costs obtained with fleets ST and ST&SP and SP and ST&SP (all p-values < 0.008). In contrast, no differences were found between the costs obtained with fleets ST and SP (all p-values > 0.84).

Based on the results of the statistical tests, we can say that the most convenient alternative in this scenario is to rent mixed fleets of special and standard chassis. This finding indicates that the rental rates of special chassis are so high, compared with those of standard chassis, that the advantages of the special chassis (regarding container compatibility and load capacity) are only materialized when these chassis are rented jointly with standard chassis to reduce costs. On the other hand, renting only special chassis voids the benefits of these chassis regarding container compatibility and load capacity, as the resulting costs are akin to those of renting only standard chassis. These results are consistent regardless of the number of service requests and the distances of the drayage movements. Nonetheless, as one may expect, higher costs and larger fleets must be considered when the number of requests increases and the distances involved in the routing are longer.

6. Conclusions

We proposed a generic modeling approach for drayage routing problems involving containers of any size and cargo category and heterogeneous trucks. This approach is based on a state transition logic that allows for the consideration of truck/container

compatibility constraints and truck load configurations. Based on this logic, we developed two mixed-integer programming models: one is solved with a black-box solver, the other with a tailored branch-and-cut algorithm. Considering problem instances with different numbers of requests, geographical distributions of locations, time-window lengths, and fleets, we conducted an extensive set of computational experiments to stress-test the limits of the problems that can be solved with each model. We analyzed how the characteristics of an instance determine its difficulty. In doing so, we ranked the instance characteristics according to their impact on the solution process. In addition, we used the tailored branch-and-cut method, which can solve instances with up to 1000 requests, to obtain optimal fleet compositions in three different scenarios of fleet decision-making.

Further research on this topic can use the proposed state-transition logic to model drayage problems involving empty container repositioning and decisions regarding the separation between trucks and containers along the routes.

CRediT authorship contribution statement

Daniel Bustos-Coral: Conceptualization, Investigation, Methodology, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. **Alysson M. Costa:** Conceptualization, Supervision, Resources, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

We thank Mr. Cameron Patrick from the Statistical Consulting Centre at The University of Melbourne for his assistance with the statistical analyses. We also thank the editor and the four reviewers for their numerous comments and insightful suggestions that have helped improve both the content and the presentation of the paper.

Funding

The first author is funded by a Melbourne Research Scholarship granted by the University of Melbourne. In addition, this research was partially funded by the Australian Government through the Australian Research Council Industrial Transformation Training Centre in Optimisation Technologies, Integrated Methodologies, and Applications (OPTIMA), Project ID IC200100009.

Appendix. Detailed solutions

We report the solutions as follows: For proven optimal solutions, we use the format "*objective function value|runtime". For solutions returned by the solver after reaching the runtime limit, we use the format "upper bound|lower bound|gap|runtime limit". The upper bound and gap are denoted by "N/A" for the instances unsolved within the runtime limit.

Tables A.9–A.11 show the solutions found for the instances with 25, 50, and 75 requests. The tables present the solutions according to the instances' characteristics: the rows indicate the geographical distribution of nodes and fleet, while the columns indicate the type of time windows. The solutions found by the fragment-based approach and the three-index MIP formulation are displayed in the rows marked as "fragments" and "three-index", respectively. The solutions for instances with 100 to 1000 requests are shown in Tables A.12 and A.13.

Table A.9
Solutions to 25-request instances.

			W1	W2	W3
C1	F1	fragments	*3822.73|0.05 s	*3213.59|0.25 s	*3310.17|0.05 s
		three-index	*3822.73|0.96 s	*3213.59|1.30 s	*3310.17|1.17 s
	F2	fragments	*3191.12|0.07 s	*2847.48|0.09 s	*2749.36|0.06 s
		three-index	*3191.12|50.22 s	2847.48|2466.69|13.4%|1 h	*2749.36|152.01 s
	F3	fragments	*2916.66|0.03 s	*2371.66|0.23 s	*3262.49|0.13 s
		three-index	*2916.66|1.83 s	*2371.66|67.76 s	*3262.49|1.02 s
	F4	fragments	*3074.82|0.05 s	*3481.05|0.09 s	*3148.65|0.09 s
		three-index	*3074.82|0.34 s	*3481.05|0.60 s	*3148.65|0.41 s
	F5	fragments	*3730.85|0.03 s	*3816.44|0.08 s	*4008.74|0.04 s
		three-index	*3730.85|0.24 s	*3816.44|0.28 s	*4008.74|0.27 s

(*continued on next page*)

Table A.9 (*continued*).

			W1	W2	W3
C2	F1	fragments	*2249.83|0.04 s	*1381.32|0.98 s	*2087.49|0.35 s
		three-index	*2249.83|1.41 s	*1381.32|18.01 s	*2087.49|1.98 s
	F2	fragments	*1617.75|0.24 s	*1128.95|0.91 s	*1476.57|0.09 s
		three-index	*1617.75|78.08 s	*1128.95|979.41 s	*1476.57|136.11 s
	F3	fragments	*1021.25|0.10 s	*866.57|0.16 s	*1348.23|0.05 s
		three-index	*1021.25|6.75 s	*866.57|30.43 s	*1348.23|6.62 s
	F4	fragments	*1345.67|0.08 s	*1254.44|0.17 s	*1685.85|0.17 s
		three-index	*1345.67|0.83 s	*1254.44|1.22 s	*1685.85|0.65 s
	F5	fragments	*1979.10|0.03 s	*1567.11|0.25 s	*1801.29|0.04 s
		three-index	*1979.10|0.20 s	*1567.11|0.17 s	*1801.29|0.23 s
R	F1	fragments	*2442.02|0.05 s	*1867.33|0.26 s	*2525.41|0.07 s
		three-index	*2442.02|1.32 s	*1867.33|106.66 s	*2525.41|1.16 s
	F2	fragments	*2578.29|0.09 s	*1954.67|0.90 s	*1945.18|0.03 s
		three-index	*2578.29|17.27 s	*1954.67|226.10 s	*1945.18|1129.10 s
	F3	fragments	*2274.37|0.04 s	*1748.13|0.14 s	*2209.90|0.25 s
		three-index	*2274.37|2.16 s	*1748.13|46.65 s	*2209.90|1.93 s
	F4	fragments	*2413.87|0.12 s	*2022.11|0.06 s	*2575.47|0.07 s
		three-index	*2413.87|0.72 s	*2022.11|0.41 s	*2575.47|0.55 s
	F5	fragments	*2872.03|0.03 s	*2718.45|0.05 s	*2685.01|0.04 s
		three-index	*2872.03|0.28 s	*2718.45|0.22 s	*2685.01|0.22 s
RC1	F1	fragments	*2839.73|0.03 s	*2542.20|0.23 s	*3261.44|0.08 s
		three-index	*2839.73|1.16 s	*2542.20|2.28 s	*3261.44|1.40 s
	F2	fragments	*2680.55|0.04 s	*2388.00|0.10 s	*2603.64|0.09 s
		three-index	*2680.55|81.78 s	*2388.00|1438.32 s	2603.64|2390.72|8.2%|1 h
	F3	fragments	*2418.69|0.05 s	*1907.49|0.99 s	*2336.21|0.19 s
		three-index	*2418.69|1.19 s	1907.49|1858.88|2.5%|1 h	*2336.21|46.61 s
	F4	fragments	*3053.76|0.05 s	*2387.88|0.22 s	*2502.64|0.06 s
		three-index	*3053.76|0.43 s	*2387.88|0.44 s	*2502.64|0.42 s
	F5	fragments	*3036.11|0.03 s	*3291.84|0.08 s	*2808.28|0.08 s
		three-index	*3036.11|0.23 s	*3291.84|0.25 s	*2808.28|0.35 s
RC2	F1	fragments	*2211.47|0.09 s	*1577.14|0.37 s	*1690.89|0.06 s
		three-index	*2211.47|0.92 s	*1577.14|135.89 s	*1690.89|1.24 s
	F2	fragments	*1560.37|0.14 s	*1482.08|0.40 s	*1826.83|0.12 s
		three-index	*1560.37|134.19 s	*1482.08|167.82 s	*1826.83|243.12 s
	F3	fragments	*1977.97|0.03 s	*1352.75|1.25 s	*1882.28|0.07 s
		three-index	*1977.97|0.78 s	*1352.75|191.06 s	*1882.28|3.88 s
	F4	fragments	*2025.23|0.05 s	*1759.51|0.06 s	*2568.22|0.07 s
		three-index	*2025.23|0.42 s	*1759.51|0.47 s	*2568.22|0.68 s
	F5	fragments	*2357.25|0.03 s	*2115.09|0.05 s	*2452.36|0.04 s
		three-index	*2357.25|0.56 s	*2115.09|0.20 s	*2452.36|0.24 s

Table A.10
Solutions to 50-request instances.

			W1	W2	W3
C1	F1	fragments	*7126.96|0.11 s	*5885.96|0.57 s	*6642.58|0.16 s
		three-index	*7126.96|8.47 s	*5885.96|75.04 s	*6642.58|8.34 s
	F2	fragments	*5183.97|0.45 s	*4969.69|16.00 s	*5281.33|0.65 s
		three-index	N/A|4510.41|N/A|1 h	N/A|3088.53|N/A|1 h	N/A|3853.47|N/A|1 h
	F3	fragments	*5659.12|0.33 s	*4343.16|1.17 s	*5339.69|0.39 s
		three-index	*5659.12|11.77 s	N/A|3207.00|N/A|1 h	*5339.69|2674.16 s
	F4	fragments	*6000.15|0.20 s	*5548.68|0.59 s	*6309.69|0.15 s
		three-index	*6000.15|6.19 s	*5548.68|6.13 s	*6309.69|5.99 s
	F5	fragments	*6896.80|0.38 s	*6524.63|0.32 s	*6965.38|0.10 s
		three-index	*6896.80|3.41 s	*6524.63|2.59 s	*6965.38|2.53 s

(*continued on next page*)

Table A.10 (*continued*).

			W1	W2	W3
C2	F1	fragments	*3712.82|0.11 s	*2650.20|1.50 s	*3709.76|0.21 s
		three-index	*3712.82|7.51 s	2650.20|2507.53|5.4%|1 h	*3709.76|23.09 s
	F2	fragments	*2689.69|0.97 s	*2400.93|29.90 s	*2688.36|1.51 s
		three-index	N/A|2358.12|N/A|1 h	N/A|1896.04|N/A|1 h	N/A|2294.10|N/A|1 h
	F3	fragments	*1957.43|1.23 s	*2162.42|5.74 s	*2070.43|2.40 s
		three-index	1957.43|1921.04|1.9%|1 h	2167.55|2068.26|4.6%|1 h	2235.96|1881.46|15.9%|1 h
	F4	fragments	*2726.05|0.16 s	*2124.05|4.54 s	*3050.40|2.21 s
		three-index	*2726.05|12.92 s	2124.05|2042.14|3.9%|1 h	*3050.40|1462.44 s
	F5	fragments	*3892.18|0.08 s	*2883.35|11.51 s	*3883.98|0.22 s
		three-index	*3892.18|2.21 s	*2883.35|199.70 s	*3883.98|2.61 s
R	F1	fragments	*4735.79|0.16 s	*4521.68|1.69 s	*4577.26|0.24 s
		three-index	*4735.79|10.29 s	*4521.68|1042.01 s	*4577.26|18.73 s
	F2	fragments	*4009.86|0.54 s	*3391.27|45.57 s	*3419.95|8.94 s
		three-index	N/A|3527.24|N/A|1 h	N/A|2001.84|N/A|1 h	N/A|2363.64|N/A|1 h
	F3	fragments	*3760.94|0.11 s	*3544.36|4.97 s	*3816.21|0.51 s
		three-index	*3760.94|22.22 s	N/A|2696.36|N/A|1 h	*3816.21|371.95 s
	F4	fragments	*5021.79|0.14 s	*4069.09|1.34 s	*3919.53|0.47 s
		three-index	*5021.79|7.04 s	*4069.09|120.47 s	*3919.53|32.27 s
	F5	fragments	*6097.80|0.12 s	*5158.57|0.20 s	*5534.97|0.26 s
		three-index	*6097.80|3.57 s	*5158.57|6.43 s	*5534.97|5.22 s
RC1	F1	fragments	*5905.51|0.11 s	*4865.30|0.71 s	*5155.86|0.24 s
		three-index	*5905.51|8.77 s	*4865.30|2130.21 s	*5155.86|12.40 s
	F2	fragments	*4843.99|0.53 s	*3923.67|3.08 s	*4934.67|1.19 s
		three-index	N/A|3411.54|N/A|1 h	N/A|2593.08|N/A|1 h	N/A|3564.20|N/A|1 h
	F3	fragments	*4290.70|0.06 s	*4381.43|0.49 s	*4885.38|0.17 s
		three-index	*4290.70|238.05 s	*4381.43|714.73 s	4885.38|4670.93|4.4%|1 h
	F4	fragments	*5792.97|0.13 s	*5443.72|1.86 s	*5121.86|0.37 s
		three-index	*5792.97|7.04 s	*5443.72|140.01 s	*5121.86|8.14 s
	F5	fragments	*6133.80|0.08 s	*6188.12|0.15 s	*6145.94|0.29 s
		three-index	*6133.80|3.10 s	*6188.12|8.95 s	*6145.94|15.73 s
RC2	F1	fragments	*3338.58|1.02 s	*2933.51|6.48 s	*3778.22|0.55 s
		three-index	*3338.58|19.56 s	2936.09|2719.54|7.4%|1 h	*3778.22|17.07 s
	F2	fragments	*3728.45|0.98 s	*3074.53|4.07 s	*3466.96|1.41 s
		three-index	N/A|3145.70|N/A|1 h	N/A|2106.81|N/A|1 h	N/A|3101.87|N/A|1 h
	F3	fragments	*3019.86|0.44 s	*2669.49|3.16 s	*3143.39|2.16 s
		three-index	*3019.86|647.48 s	N/A|2487.11|N/A|1 h	N/A|2884.15|N/A|1 h
	F4	fragments	*3899.26|0.19 s	*3225.37|6.41 s	*3343.08|1.69 s
		three-index	*3899.26|9.91 s	N/A|2959.39|N/A|1 h	*3343.08|2519.36 s
	F5	fragments	*4551.39|0.08 s	*4161.35|0.30 s	*4085.26|0.28 s
		three-index	*4551.39|4.36 s	*4161.35|5.65 s	*4085.26|7.03 s

Table A.11
Solutions to 75-request instances.

			W1	W2	W3
C1	F1	fragments	*9972.16|0.23 s	*8574.80|1.30 s	*10,077.00|0.41 s
		three-index	*9972.16|44.67 s	*8574.80|153.50 s	*10,077.00|65.20 s
	F2	fragments	*8091.92|1.16 s	*7647.68|8.52 s	*7503.23|10.32 s
		three-index	N/A|2240.56|N/A|1 h	N/A|1001.57|N/A|1 h	N/A|1412.74|N/A|1 h
	F3	fragments	*8363.74|0.92 s	*7292.02|4.81 s	*8002.06|1.65 s
		three-index	8363.74|8305.09|0.7%|1 h	N/A|5776.60|N/A|1 h	8002.06|7626.25|4.7%|1 h
	F4	fragments	*9714.16|0.19 s	*7955.52|1.77 s	*9170.22|0.83 s
		three-index	*9714.16|16.51 s	*7955.52|3590.29 s	*9170.22|21.31 s
	F5	fragments	*10,256.36|0.14 s	*9949.94|0.42 s	*10,320.45|0.52 s
		three-index	*10,256.36|14.80 s	*9949.94|17.03 s	*10,320.45|12.36 s

(*continued on next page*)

Table A.11 (*continued*).

			W1	W2	W3
C2	F1	fragments	*5164.12|0.48 s	*3750.81|4.62 s	*5166.93|3.17 s
		three-index	*5164.12|170.29 s	N/A|3444.26|N/A|1 h	*5166.93|1082.26 s
	F2	fragments	*4303.94|2.69 s	*2830.49|1457.14 s	*3509.35|34.79 s
		three-index	N/A|1194.64|N/A|1 h	N/A|1683.24|N/A|1 h	N/A|870.57|N/A|1 h
	F3	fragments	*3512.74|2.12 s	*2907.52|807.74 s	*3873.44|65.07 s
		three-index	N/A|3315.35|N/A|1 h	N/A|1863.34|N/A|1 h	N/A|3119.95|N/A|1 h
	F4	fragments	*4982.08|0.77 s	*3450.25|538.11 s	*4340.95|10.48 s
		three-index	*4982.08|73.78 s	N/A|3114.87|N/A|1 h	4368.81|4222.67|3.3%|1 h
	F5	fragments	*5139.03|0.16 s	*4515.02|3.28 s	*4451.20|0.48 s
		three-index	*5139.03|9.77 s	*4515.02|88.99 s	*4451.20|12.32 s
R	F1	fragments	*7234.15|0.26 s	*6392.45|5.12 s	*6990.10|0.93 s
		three-index	*7234.15|66.12 s	N/A|5268.75|N/A|1 h	*6990.10|166.59 s
	F2	fragments	*5633.17|0.96 s	*5026.04|97.37 s	*5674.50|2.96 s
		three-index	N/A|4469.49|N/A|1 h	N/A|466.58|N/A|1 h	N/A|731.39|N/A|1 h
	F3	fragments	*5764.07|0.63 s	*5785.22|8.83 s	*5428.62|1.78 s
		three-index	5815.34|5619.97|3.4%|1 h	N/A|4533.71|N/A|1 h	N/A|4937.99|N/A|1 h
	F4	fragments	*7255.03|0.20 s	*5612.36|5.84 s	*6175.91|4.02 s
		three-index	*7255.03|16.71 s	*5612.36|2116.88 s	*6175.91|576.66 s
	F5	fragments	*7051.56|0.50 s	*6479.30|1.78 s	*7551.07|0.31 s
		three-index	*7051.56|11.27 s	*6479.30|45.79 s	*7551.07|42.60 s
RC1	F1	fragments	*7894.95|1.65 s	*6931.35|4.44 s	*7738.40|0.77 s
		three-index	*7894.95|115.51 s	N/A|5075.34|N/A|1 h	*7738.40|77.67 s
	F2	fragments	*6868.00|0.24 s	*5880.55|109.92 s	*6960.89|4.45 s
		three-index	N/A|4858.65|N/A|1 h	N/A|943.32|N/A|1 h	N/A|1605.81|N/A|1 h
	F3	fragments	*7448.88|0.58 s	*6226.97|1.50 s	*7445.36|0.18 s
		three-index	*7448.88|961.96 s	N/A|5079.04|N/A|1 h	N/A|7003.46|N/A|1 h
	F4	fragments	*8346.91|0.35 s	*7011.78|1.93 s	*8166.84|0.85 s
		three-index	*8346.91|21.28 s	7011.78|6557.78|6.5%|1 h	*8166.84|37.98 s
	F5	fragments	*9567.19|0.15 s	*7894.69|0.44 s	*8702.35|0.11 s
		three-index	*9567.19|12.76 s	*7894.69|12.77 s	*8702.35|11.45 s
RC2	F1	fragments	*5624.80|0.27 s	*4872.05|23.62 s	*6240.76|2.95 s
		three-index	*5624.80|101.76 s	N/A|4036.86|N/A|1 h	*6240.76|368.72 s
	F2	fragments	*4682.78|1.08 s	*3670.98|466.82 s	*4498.51|74.19 s
		three-index	N/A|1151.15|N/A|1 h	N/A|375.95|N/A|1 h	N/A|652.35|N/A|1 h
	F3	fragments	*5399.58|1.42 s	*4035.37|32.13 s	*4231.39|5.00 s
		three-index	N/A|5243.54|N/A|1 h	N/A|2843.97|N/A|1 h	N/A|3682.56|N/A|1 h
	F4	fragments	*5700.68|1.36 s	*3975.52|17.44 s	*5102.66|3.62 s
		three-index	*5700.68|26.92 s	N/A|3063.12|N/A|1 h	5102.66|4977.53|2.5%|1 h
	F5	fragments	*7108.35|0.43 s	*5340.85|2.51 s	*6164.23|1.23 s
		three-index	*7108.35|11.40 s	*5340.85|1198.44 s	*6164.23|737.06 s

Table A.12
Solutions to 100-, 200-, and 400-request instances.

Geo. dist.	Fleet	Time win.	No. of requests		
			100	200	400
C1	F1	W1	*13,167.62|0.43 s	*26,308.12|1.88 s	*100,878.30|4.48 s
C1	F1	W2	*10,299.24|6.03 s	*21,519.60|63.90 s	*92,083.91|11.10 s
C1	F1	W3	*13,103.54|0.71 s	*24,612.81|5.33 s	*97,219.83|8.05 s
C1	F2	W1	*10,289.93|2.26 s	*20,497.12|42.62 s	*78,777.33|52.11 s
C1	F2	W2	*9218.26|222.41 s	–	*73,121.64|100.36 s
C1	F2	W3	*11,078.75|4.94 s	*19,132.80|1671.92 s	*77,089.84|29.07 s

(*continued on next page*)

Table A.12 (*continued*).

Geo. dist.	Fleet	Time win.	No. of requests		
			100	200	400
C1	F3	W1	*10,082.75|1.16 s	*19,925.17|12.48 s	*79,554.12|12.09 s
C1	F3	W2	*9051.40|6.17 s	*18,124.45|424.74 s	*71,737.09|49.36 s
C1	F3	W3	*10,366.43|5.00 s	*18,394.33|335.42 s	*77,280.01|25.52 s
C1	F4	W1	*11,689.28|0.59 s	*23,192.29|2.16 s	*94,724.62|7.24 s
C1	F4	W2	*9963.80|12.47 s	*21,034.21|110.57 s	*86,327.86|21.03 s
C1	F4	W3	*11,980.50|0.24 s	*21,802.87|24.61 s	*91,617.46|17.71 s
C1	F5	W1	*13,917.66|0.26 s	*26,848.67|1.25 s	*99,333.33|1.57 s
C1	F5	W2	*12,479.70|1.27 s	*24,603.00|5.82 s	*96,283.92|6.49 s
C1	F5	W3	*13,488.29|1.05 s	*26,931.78|3.02 s	*98,889.02|16.56 s
C2	F1	W1	*7272.15|0.69 s	*12,861.01|8.97 s	*35,100.31|837.15 s
C2	F1	W2	*4755.32|52.01 s	*9269.92|1366.64 s	69,362.20|24,470.82|64.7%|2 h
C2	F1	W3	*5623.07|7.81 s	*10,650.72|158.48 s	*36,002.14|682.77 s
C2	F2	W1	*5246.78|22.77 s	9524.20|9405.80|1.2%|2 h	31,659.41|31,474.11|0.6%|2 h
C2	F2	W2	4187.20|4023.13|3.9%|2 h	–	–
C2	F2	W3	*5067.08|330.10 s	–	–
C2	F3	W1	*4223.91|2.78 s	*9112.25|92.26 s	*25,344.88|2259.12 s
C2	F3	W2	*3336.51|2762.74 s	–	–
C2	F3	W3	*3866.65|114.01 s	–	–
C2	F4	W1	*4989.31|6.80 s	*9894.89|165.97 s	*31,258.25|862.97 s
C2	F4	W2	*3876.19|282.67 s	–	–
C2	F4	W3	*6164.55|11.92 s	8917.54|8830.02|1.0%|2 h	29,495.52|29,190.25|1.03%|2 h
C2	F5	W1	*6763.07|0.70 s	*13,388.58|5.66 s	*36,438.97|26.72 s
C2	F5	W2	*5064.23|7.42 s	*9961.68|265.45 s	27,338.92|26,527.43|3.0%|2 h
C2	F5	W3	*6849.01|0.58 s	*12,605.91|12.23 s	*34,000.07|138.34 s
R	F1	W1	*8875.48|0.91 s	*17,953.98|20.05 s	*67,430.53|49.61 s
R	F1	W2	*7892.34|12.54 s	*15,439.26|180.63 s	*51,622.24|583.28 s
R	F1	W3	*8033.33|1.17 s	*18,767.86|12.58 s	*66,456.92|40.62 s
R	F2	W1	*7383.16|5.68 s	*14,953.01|271.33 s	*49,436.68|84.96 s
R	F2	W2	*6528.31|3830.88 s	–	–
R	F2	W3	*7279.59|69.45 s	–	–
R	F3	W1	*7518.07|2.46 s	*14,376.46|33.87 s	*52,728.84|15.05 s
R	F3	W2	*5945.51|22.23 s	12,760.77|12,662.53|0.8%|2 h	–
R	F3	W3	*7681.13|7.40 s	*13,675.74|75.67 s	*52,629.61|40.39 s
R	F4	W1	*9839.41|0.95 s	*16,637.54|23.34 s	*57,474.31|17.70 s
R	F4	W2	*7807.85|14.72 s	*13,755.98|266.79 s	*51,700.28|166.53 s
R	F4	W3	*8367.21|1.08 s	*15,010.38|27.80 s	*52,433.24|22.24 s
R	F5	W1	*10,349.42|0.26 s	*19,086.32|0.38 s	*69,326.24|8.75 s
R	F5	W2	*8992.71|1.72 s	*16,516.83|20.31 s	*59,340.52|13.25 s
R	F5	W3	*10,491.07|0.99 s	*18,693.53|6.22 s	*60,672.87|18.67 s
RC1	F1	W1	*10,521.10|0.44 s	*21,424.71|2.92 s	*80,881.55|6.00 s
RC1	F1	W2	*8919.95|23.79 s	*17,183.74|219.71 s	*71,958.22|62.81 s
RC1	F1	W3	*10,226.99|1.57 s	*20,642.63|19.91 s	*77,057.01|16.58 s
RC1	F2	W1	*9480.82|2.64 s	*16,779.11|68.53 s	*64,482.08|16.77 s
RC1	F2	W2	*8444.50|213.59 s	–	–
RC1	F2	W3	*8587.19|79.76 s	*16,625.42|2261.99 s	*61,448.91|134.52 s
RC1	F3	W1	*9547.80|0.45 s	*16,072.17|25.61 s	*63,677.20|8.49 s
RC1	F3	W2	*8878.10|22.00 s	*15,245.96|1297.20 s	*57,935.20|181.16 s
RC1	F3	W3	*8825.13|6.34 s	*15,961.62|41.78 s	*64,055.75|41.45 s
RC1	F4	W1	*10,971.86|0.16 s	*19,897.86|1.30 s	*76,370.54|10.01 s
RC1	F4	W2	*9125.85|1.42 s	*17,477.35|87.62 s	*65,344.44|26.04 s
RC1	F4	W3	*9476.36|3.34 s	*19,947.67|15.68 s	*74,678.96|14.92 s
RC1	F5	W1	*11,917.92|0.26 s	*24,171.55|0.84 s	*81,419.10|10.86 s
RC1	F5	W2	*9802.43|2.69 s	*18,635.93|3.49 s	*71,676.21|10.04 s
RC1	F5	W3	*12,156.22|0.58 s	*22,700.51|3.85 s	*81,465.32|9.83 s
RC2	F1	W1	*8149.36|1.06 s	*14,164.86|57.53 s	*51,456.69|47.18 s
RC2	F1	W2	*6660.29|13.34 s	*12,267.72|4180.81 s	*41,734.79|2652.57 s
RC2	F1	W3	*7892.28|10.79 s	*13,121.45|98.95 s	*49,437.28|154.96 s
RC2	F2	W1	*6436.65|26.50 s	*11,409.49|571.29 s	*41,390.60|319.96 s
RC2	F2	W2	4957.95|4812.74|2.9%|2 h	–	–
RC2	F2	W3	*7117.48|453.42 s	–	–
RC2	F3	W1	*6046.38|6.37 s	*11,911.12|61.91 s	*36,680.93|135.57 s
RC2	F3	W2	*4953.33|160.96 s	–	–
RC2	F3	W3	*5905.17|20.69 s	*10,513.46|765.56 s	*40,867.00|7139.90 s

(*continued on next page*)

Table A.12 (*continued*).

Geo. dist.	Fleet	Time win.	No. of requests		
			100	200	400
RC2	F4	W1	*6387.00|4.18 s	*13,188.32|12.20 s	*44,492.27|427.72 s
RC2	F4	W2	*5991.95|148.64 s	10,943.15|10,919.34|0.2%|2 h	37,806.44|37,507.79|0.8%|2 h
RC2	F4	W3	*6456.95|9.02 s	*12,214.98|1317.62 s	*49,062.50|243.86 s
RC2	F5	W1	*8083.44|1.13 s	*14,988.72|9.95 s	*55,569.37|13.97 s
RC2	F5	W2	*6931.14|13.12 s	*12,302.96|164.72 s	39,431.99|39,369.98|0.2%|2 h
RC2	F5	W3	*8166.60|0.75 s	*15,923.24|5.54 s	*48,659.96|87.58 s

Table A.13
Solutions to 600-, 800-, and 1000-request instances.

Geo. dist.	Fleet	Time win.	No. of requests		
			600	800	1000
C1	F1	W1	*151,193.69|10.34 s	*199,727.02|18.90 s	*250,554.14|29.72 s
C1	F1	W2	*140,151.42|25.03 s	*185,749.30|47.37 s	*228,100.54|78.73 s
C1	F1	W3	*149,418.84|19.02 s	*195,183.78|31.97 s	*247,647.03|52.61 s
C1	F2	W1	*117,918.91|71.70 s	*152,770.84|124.09 s	–
C1	F2	W2	–	–	–
C1	F2	W3	–	–	–
C1	F3	W1	*120,220.28|39.09 s	*153,865.51|49.20 s	*196,862.39|255.79 s
C1	F3	W2	*110,330.33|133.62 s	–	–
C1	F3	W3	*113,517.78|84.70 s	*156,578.61|116.55 s	–
C1	F4	W1	*138,129.22|27.30 s	*182,997.98|22.07 s	*222,757.00|43.18 s
C1	F4	W2	*127,533.40|66.64 s	*167,607.67|113.98 s	*213,117.28|166.25 s
C1	F4	W3	*137,586.88|21.89 s	*184,187.76|36.53 s	*226,947.43|37.23 s
C1	F5	W1	*152,150.51|14.39 s	*197,677.78|24.01 s	*244,287.32|36.55 s
C1	F5	W2	*140,227.76|46.10 s	*185,780.25|45.51 s	*230,263.30|276.80 s
C1	F5	W3	*147,670.89|36.28 s	*195,517.29|52.99 s	*248,025.92|229.93 s
C2	F1	W1	*50,593.56|2330.59 s	*75,706.34|5615.83 s	80,038.55|79,225.35|1.02%|2 h
C2	F1	W2	105,433.18|35,852.41|66.0%|2 h	131,291.98|45,790.35|65.1%|2 h	163,053.77|56,422.48|65.4%|2 h
C2	F1	W3	45,796.74|45,400.39|0.9%|2 h	142,558.27|68,244.59|52.1%|2 h	171,437.54|81,071.39|52.7%|2 h
C2	F2	W1	–	–	–
C2	F2	W2	–	–	–
C2	F2	W3	–	–	–
C2	F3	W1	–	–	–
C2	F3	W2	–	–	–
C2	F3	W3	–	–	–
C2	F4	W1	*43,996.76|4244.89 s	–	–
C2	F4	W2	–	–	–
C2	F4	W3	–	–	–
C2	F5	W1	*50,491.06|157.92 s	*72,345.56|1030.75 s	*95,781.56|612.64 s
C2	F5	W2	N/A|37,103.99|N/A|2 h	N/A|50,664.78|N/A|2 h	N/A|60,292.30|N/A|2 h
C2	F5	W3	*50,379.42|518.10 s	*69,322.20|4696.89 s	88,057.36|87,202.83|1.0%|2 h
R	F1	W1	*89,357.73|52.69 s	*124,797.93|94.23 s	*143,159.42|441.80 s
R	F1	W2	*77,867.73|1861.39 s	181,408.39|101,493.61|44.05%|2 h	223,240.50|129,278.48|42.09%|2 h
R	F1	W3	*95,654.03|76.38 s	*116,875.25|315.40 s	*150,359.88|688.58 s
R	F2	W1	–	–	–
R	F2	W2	–	–	–
R	F2	W3	–	–	–
R	F3	W1	*75,703.65|103.34 s	*108,185.46|112.55 s	–
R	F3	W2	–	–	–
R	F3	W3	–	–	–
R	F4	W1	*87,206.02|58.62 s	*117,422.52|78.73 s	*141,421.71|272.36 s
R	F4	W2	*73,917.86|1703.87 s	N/A|98,911.09|N/A|2 h	–
R	F4	W3	*92,421.35|250.30 s	*113,011.42|193.66 s	*135,866.92|619.89 s
R	F5	W1	*98,525.82|23.73 s	*135,828.27|48.51 s	*165,456.54|92.24 s
R	F5	W2	*85,412.71|152.86 s	*109,947.77|942.22 s	*145,513.94|2064.67 s
R	F5	W3	*95,332.02|64.32 s	*124,194.87|120.07 s	*149,454.74|523.88 s
RC1	F1	W1	*119,286.40|15.75 s	*157,595.97|31.12 s	*192,775.70|61.38 s
RC1	F1	W2	*108,017.04|77.00 s	*142,140.90|172.84 s	*174,269.00|432.29 s
RC1	F1	W3	*117,591.64|42.25 s	*156,061.13|84.82 s	*196,497.72|108.10 s
RC1	F2	W1	*91,318.22|72.16 s	–	–
RC1	F2	W2	–	–	–
RC1	F2	W3	–	–	–
RC1	F3	W1	*96,749.11|15.60 s	*126,117.73|227.98 s	–

(*continued on next page*)

Table A.13 (*continued*).

Geo. dist.	Fleet	Time win.	No. of requests		
			600	800	1000
RC1	F3	W2	*89,959.45|166.71 s	–	–
RC1	F3	W3	*97,147.59|108.95 s	–	–
RC1	F4	W1	*107,560.51|77.91 s	*145,051.83|97.97 s	*183,065.71|125.14 s
RC1	F4	W2	*96,987.11|137.94 s	*130,557.34|920.43 s	–
RC1	F4	W3	*111,088.43|99.84 s	*143,803.83|260.04 s	*179,090.30|152.47 s
RC1	F5	W1	*120,232.45|13.11 s	*160,340.79|28.51 s	*193,969.67|79.74 s
RC1	F5	W2	*110,695.99|44.40 s	*138,385.54|268.17 s	*177,758.23|155.67 s
RC1	F5	W3	*120,845.64|61.04 s	*160,071.54|82.95 s	*195,739.73|44.85 s
RC2	F1	W1	*77,489.00|123.52 s	*105,202.89|357.67 s	*123,381.57|2255.49 s
RC2	F1	W2	119,691.76|57,760.14|51.7%|2 h	146,140.72|75,335.69|48.4%|2 h	196,404.22|98,928.43|49.6%|2 h
RC2	F1	W3	*81,121.35|323.44 s	*96,283.49|5511.87 s	194,348.32|116,741.36|39.9%|2 h
RC2	F2	W1	–	–	–
RC2	F2	W2	–	–	–
RC2	F2	W3	–	–	–
RC2	F3	W1	*57,544.35|1587.88 s	–	–
RC2	F3	W2	–	–	–
RC2	F3	W3	–	–	–
RC2	F4	W1	*61,695.66|559.48 s	–	96,296.41|96,186.08|0.1%|2 h
RC2	F4	W2	–	–	–
RC2	F4	W3	60,550.25|60,317.86|0.4%|2 h	*84,248.90|5895.06 s	–
RC2	F5	W1	*75,357.26|120.24 s	*112,919.24|60.56 s	*118,239.98|434.55 s
RC2	F5	W2	64,029.36|63,994.29|0.05%|2 h	86,073.68|85,355.09|0.8%|2 h	113,188.36|113,136.75|0.05%|2 h
RC2	F5	W3	*74,806.92|164.19 s	*102,538.98|141.83 s	*121,824.92|1739.52 s

References

Alyasiry, A.M., Forbes, M., Bulmer, M., 2019. An exact algorithm for the pickup and delivery problem with time windows and last-in-first-out loading. Transp. Sci. 53 (6), 1695–1705.

Berbeglia, G., Cordeau, J.-F., Gribkovskaia, I., Laporte, G., 2007. Static pickup and delivery problems: A classification scheme and survey. TOP 15 (1), 1–31.

Bjelić, N., Vidović, M., Popović, D., Ratković, B., 2022. Rolling-horizon approach in solving dynamic multisize multi-trailer container drayage problem. Expert Syst. Appl. 201, 117170.

Bomboi, F., 2020. New Routing Problems with Possibly Correlated Travel Times (Ph.D. thesis). University of Cagliari, Cagliari, Italy.

Braekers, K., Caris, A., Janssens, G.K., 2013. Integrated planning of loaded and empty container movements. OR Spectrum 35 (2), 457–478.

Braekers, K., Caris, A., Janssens, G.K., 2014. Bi-objective optimization of drayage operations in the service area of intermodal terminals. Transp. Res. Part E 65, 50–69.

Bruglieri, M., Mancini, S., Peruzzini, R., Pisacane, O., 2021. The multi-period multi-trip container drayage problem with release and due dates. Comput. Oper. Res. 125, 105102.

Caris, A., Janssens, G., 2009. A local search heuristic for the pre- and end-haulage of intermodal container terminals. Comput. Oper. Res. 36 (10), 2763–2772.

Chen, R., Chen, S., Cui, H., Meng, Q., 2021a. The container drayage problem for heterogeneous trucks with multiple loads: A revisit. Transp. Res. Part E 147, 102241.

Chen, T., Guestrin, C., 2016. XGBoost: a scalable tree boosting system. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, San Francisco California USA, pp. 785–794.

Chen, R., Meng, Q., Jia, P., 2022. Container port drayage operations and management: past and future. Transp. Res. Part E 159, 102633.

Chen, S., Wang, H., Meng, Q., 2021b. Autonomous truck scheduling for container transshipment between two seaport terminals considering platooning and speed optimization. Transp. Res. B 154, 289–315.

Chung, K.H., Ko, C.S., Shin, J.Y., Hwang, H., Kim, K.H., 2007. Development of mathematical models for the container road transportation in Korean trucking industries. Comput. Ind. Eng. 53 (2), 252–262.

Cui, H., Chen, S., Chen, R., Meng, Q., 2022. A two-stage hybrid heuristic solution for the container drayage problem with trailer reposition. European J. Oper. Res. 299 (2), 468–482.

Daham, H.A., Yang, X., Warnes, M.K., 2017. An efficient mixed integer programming model for pairing containers in inland transportation based on the assignment of orders. J. Oper. Res. Soc. 68 (6), 678–694.

Desaulniers, G., Madsen, O.B., Ropke, S., 2014. Chapter 5: the vehicle routing problem with time windows. In: Toth, P., Vigo, D. (Eds.), Vehicle Routing: Problems, Methods, and Applications, second ed. Society for Industrial and Applied Mathematics, Philadelphia, PA, pp. 119–159.

Dumas, Y., Desrosiers, J., Soumis, F., 1991. The pickup and delivery problem with time windows. European J. Oper. Res. 54 (1), 7–22.

Escudero-Santana, A., Muñuzuri, J., Cortés, P., Onieva, L., 2020. The one container drayage problem with soft time windows. Res. Transp. Econ. 100884.

Flexi-Van Leasing LLC, 2022. Daily rentals. URL: https://flexivan.com/daily-rentals/. Accessed: 2022-02-26.

Francis, P., Zhang, G., Smilowitz, K., 2007. Improved modeling and solution methods for the multi-resource routing problem. European J. Oper. Res. 180 (3), 1045–1059.

Funke, J., Kopfer, H., 2016. A model for a multi-size inland container transportation problem. Transp. Res. Part E 89, 70–85.

Ghezelsoflu, A., Di Francesco, M., Frangioni, A., Zuddas, P., 2018. A set-covering formulation for a drayage problem with single and double container loads. J. Ind. Eng. Int. 14 (4), 665–676.

Ghezelsoflu, A., Di Francesco, M., Frangioni, A., Zuddas, P., 2021. A multiperiod drayage problem with customer-dependent service periods. Comput. Oper. Res. 129, 105189.

Giuliano, G., Dessouky, M., Dexter, S., Fang, J., Hu, S., Miller, M., 2021. Heavy-duty trucks: the challenge of getting to zero. Transp. Res. Part D 93, 102742.

Golden, B., Assad, A., Levy, L., Gheysens, F., 1984. The fleet size and mix vehicle routing problem. Comput. Oper. Res. 11 (1), 49–66.

Gronalt, M., Hartl, R.F., Reimann, M., 2003. New savings based algorithms for time constrained pickup and delivery of full truckloads. European J. Oper. Res. 151 (3), 520–535.

Heggen, H., Molenbruch, Y., Caris, A., Braekers, K., 2019. Intermodal container routing: integrating long-haul routing and local drayage decisions. Sustainability 11 (6), 1634.
Ileri, Y., Bazaraa, M., Gifford, T., Nemhauser, G., Sokol, J., Wikum, E., 2006. An optimization approach for planning daily drayage operations. CEJOR Cent. Eur. J. Oper. Res. 14 (2), 141–156.
Imai, A., Nishimura, E., Current, J., 2007. A Lagrangian relaxation-based heuristic for the vehicle routing with full container load. European J. Oper. Res. 176 (1), 87–105.
Jula, H., Dessouky, M., Ioannou, P., Chassiakos, A., 2005. Container movement by trucks in metropolitan networks: Modeling and optimization. Transp. Res. Part E 41 (3), 235–259.
Lai, M., Battarra, M., Di Francesco, M., Zuddas, P., 2015. An adaptive guidance meta-heuristic for the vehicle routing problem with splits and clustered backhauls. J. Oper. Res. Soc. 66 (7), 1222–1235.
Lai, M., Crainic, T.G., Di Francesco, M., Zuddas, P., 2013. An heuristic search for the routing of heterogeneous trucks with single and double container loads. Transp. Res. Part E 56, 108–118.
Lundberg, S.M., Erion, G., Chen, H., DeGrave, A., Prutkin, J.M., Nair, B., Katz, R., Himmelfarb, J., Bansal, N., Lee, S.-I., 2020. From local explanations to global understanding with explainable AI for trees. Nat. Mach. Intell. 2 (1), 56–67.
Lundberg, S.M., Lee, S.-I., 2017. A unified approach to interpreting model predictions. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (Eds.), Advances in Neural Information Processing Systems, Vol. 30. Curran Associates, Inc., Long Beach, CA, p. 10.
Máhr, T., Srour, J., de Weerdt, M., Zuidwijk, R., 2010. Can agents measure up? A comparative study of an agent-based and on-line optimization approach for a drayage problem with uncertainty. Transp. Res. C 18 (1), 99–119.
Max-Atlas Equipment International Inc, 2022. Create your own container chassis that fits your needs. URL: https://max-atlas.com/container-chassis/. Accessed: 2022-05-14.
Meisel, F., Kopfer, H., 2014. Synchronized routing of active and passive means of transport. OR Spectrum 36 (2), 297–322.
Moghaddam, M., Pearce, R.H., Mokhtar, H., Prato, C.G., 2020. A generalised model for container drayage operations with heterogeneous fleet, multi-container sizes and two modes of operation. Transp. Res. Part E 139, 101973.
Namboothiri, R., Erera, A.L., 2008. Planning local container drayage operations given a port access appointment system. Transp. Res. Part E 44 (2), 185–202.
Nossack, J., Pesch, E., 2013. A truck scheduling problem arising in intermodal container transportation. European J. Oper. Res. 230 (3), 666–680.
Pazour, J.A., Neubert, L.C., 2013. Routing and scheduling of cross-town drayage operations at j.b. hunt transport. Interfaces 43 (2), 117–129.
Pérez Rivera, A.E., Mes, M.R.K., 2017. Scheduling drayage operations in synchromodal transport. In: Bektaş, T., Coniglio, S., Martinez-Sykora, A., Voß, S. (Eds.), Computational Logistics. Springer Berlin Heidelberg, New York, NY, pp. 404–419.
Popović, D., Bjelić, N., Vidović, M., 2018. Containers drayage problem with simultaneous routing of vehicles and handling equipment. In: Proceedings of the XIII Balkan Conference on Operational Research. Belgrade, Serbia, pp. 142–149.
Popović, D., Vidović, M., Nikolić, M., 2014. The variable neighborhood search heuristic for the containers drayage problem with time windows. In: Snášel, V., Krömer, P., Köppen, M., Schaefer, G. (Eds.), Soft Computing in Industrial Applications. Springer International Publishing, pp. 351–364.
Ray, S., Soeanu, A., Berger, J., Debbabi, M., 2014. The multi-depot split-delivery vehicle routing problem: model and solution algorithm. Knowl.-Based Syst. 71, 238–265.
Regnier-Coudert, O., McCall, J., Ayodele, M., Anderson, S., 2016. Truck and trailer scheduling in a real world, dynamic and heterogeneous context. Transp. Res. Part E 93, 389–408.
Reinhardt, L.B., Pisinger, D., Spoorendonk, S., Sigurd, M.M., 2016. Optimization of the drayage problem using exact methods. INFOR: Inform. Syst. Oper. Res. 54 (1), 33–51.
Ritzinger, U., Hu, B., Koller, H., Dragaschnig, M., 2017. Synchronizing trucks and trailers in a multiresource problem with variable neighborhood search. Transp. Res. Record: J. Transp. Res. Board 2610 (1), 28–34.
Schulte, F., Lalla-Ruiz, E., González-Ramírez, R.G., Voß, S., 2017. Reducing port-related empty truck emissions: A mathematical approach for truck appointments with collaboration. Transp. Res. Part E 105, 195–212.
Shapley, L.S., 1953. A value for N-person games. In: Contributions to the Theory of Games, Vol. 28. pp. 307–317.
Shiri, S., Huynh, N., 2016. Optimization of drayage operations with time-window constraints. Int. J. Prod. Econ. 176, 7–20.
Shiri, S., Huynh, N., 2018. Assessment of U.S. Chassis supply models on drayage productivity and air emissions. Transp. Res. Part D 61, 174–203.
Smilowitz, K., 2006. Multi-resource routing with flexible tasks: An application in drayage operations. IIE Trans. 38 (7), 577–590.
Soeanu, A., Ray, S., Berger, J., Boukhtouta, A., Debbabi, M., 2020. Multi-depot vehicle routing problem with risk mitigation: model and solution algorithm. Expert Syst. Appl. 145, 113099.
Song, Y., Zhang, J., Liang, Z., Ye, C., 2017. An exact algorithm for the container drayage problem under a separation mode. Transp. Res. Part E 106, 231–254.
Sterzik, S., Kopfer, H., 2013. A tabu search heuristic for the inland container transportation problem. Comput. Oper. Res. 40 (4), 953–962.
Tan, K., Chew, Y., Lee, L., 2006. A hybrid multi-objective evolutionary algorithm for solving truck and trailer vehicle routing problems. European J. Oper. Res. 172 (3), 855–885.
Tilk, C., Bianchessi, N., Drexl, M., Irnich, S., Meisel, F., 2018. Branch-and-price-and-cut for the active-passive vehicle-routing problem. Transp. Sci. 52 (2), 300–319.
Vidović, M., Popović, D., Ratković, B., Radivojević, G., 2017. Generalized mixed integer and VNS heuristic approach to solving the multisize containers drayage problem. Int. Trans. Oper. Res. 24 (3), 583–614.
Vidović, M., Radivojević, G., Raković, B., 2011. Vehicle routing in containers pickup up and delivery processes. Proc. Soc. Behav. Sci. 20, 335–343.
Vijayagopal, R., Rousseau, A., 2021. Electric truck economic feasibility analysis. World Electric Veh. J. 12 (2), 75.
Wang, X., Regan, A.C., 2002. Local truckload pickup and delivery with hard time window constraints. Transp. Res. B 36 (2), 97–112.
Wang, W.F., Yun, W.Y., 2013. Scheduling for inland container truck and train transportation. Int. J. Prod. Econ. 143 (2), 349–356.
Xue, Z., Lin, W.-H., Miao, L., Zhang, C., 2015. Local container drayage problem with tractor and trailer operating in separable mode. Flexible Serv. Manufact. J. 27 (2), 431–450.
Xue, Z., Lin, H., You, J., 2021. Local container drayage problem with truck platooning mode. Transp. Res. Part E 147, 102211.
Xue, Z., Zhang, C., Lin, W.-H., Miao, L., Yang, P., 2014. A tabu search heuristic for the local container drayage problem under a new operation mode. Transp. Res. Part E 62, 136–150.
Yang, X., Daham, H.A., 2020. A column generation-based decomposition and aggregation approach for combining orders in inland transportation of containers. OR Spectrum 42 (1), 261–296.
Yang, X., Daham, H.A., Salhi, A., 2021. Combined strip and discharge delivery of containers in heterogeneous fleets with time windows. Comput. Oper. Res. 127, 105141.
You, J., Miao, L., Zhang, C., Xue, Z., 2020. A generic model for the local container drayage problem using the emerging truck platooning operation mode. Transp. Res. B 133, 181–209.
Zhang, R., Huang, C., Wang, J., 2020. A novel mathematical model and a large neighborhood search algorithm for container drayage operations with multi-resource constraints. Comput. Ind. Eng. 139, 106143.
Zhang, R., Lu, J.-C., Wang, D., 2014. Container drayage problem with flexible orders and its near real-time solution strategies. Transp. Res. Part E 61, 235–251.
Zhang, R., Wang, D., Wang, J., 2021. Multi-trailer drop-and-pull container drayage problem. IEEE Trans. Intell. Transp. Syst. 22 (9), 5708–5720.

Zhang, R., Yun, W.Y., Kopfer, H., 2010. Heuristic-based truck scheduling for inland container transportation. OR Spectrum 32 (3), 787–808.
Zhang, R., Yun, W.Y., Kopfer, H., 2015. Multi-size container transportation by truck: Modeling and optimization. Flexible Serv. Manufact. J. 27 (2), 403–430.
Zhang, R., Yun, W.Y., Moon, I., 2009. A reactive tabu search algorithm for the multi-depot container truck transportation problem. Transp. Res. Part E 45 (6), 904–914.
Zhang, R., Yun, W.Y., Moon, I.K., 2011. Modeling and optimization of a container drayage problem with resource constraints. Int. J. Prod. Econ. 133 (1), 351–359.
Zhang, R., Zhao, H., Moon, I., 2018. Range-based truck-state transition modeling method for foldable container drayage services. Transp. Res. Part E 118, 225–239.