Arnau, Q.; Barrena, E.; Panadero, J.; Torre-Martínez, MRDL.; Juan-Pérez, ÁA. (2022). A biased-randomized discrete-event heuristic for coordinated multi-vehicle container transport across interconnected networks. European Journal of Operational Research. 302(1):348-362. https://doi.org/10.1016/j.ejor.2021.12.035

Journal Name xx (2023) 1–29

A Biased-Randomized Discrete-Event Heuristic for Coordinated Multi-Vehicle Container Transport across Interconnected Networks

Quim Arnau[a], Eva Barrena[b], Javier Panadero[a,c], Rocio de la Torre[d,*] and Angel A. Juan[e]

[a]*IN3 – Computer Science Dept., Av. Carl Friedrich Gauss 5, 08860 Castelldefels, Spain*
[b]*Dep. of Economics, Quantitative Methods and Economic History, Pablo de Olavide University, 41013 Seville, Spain*
[c]*Euncet Business School, 08225 Terrassa, Spain*
[d]*Dept. of Business Organisation, Universitat Politècnica de València, 03801 Alcoy, Spain*
[e]*Dept. of Applied Statistics and Operations Research, Universitat Politècnica de València, 03801 Alcoy, Spain*
E-mail: quim.arnau@outlook.com [Q. Arnau]; ebarrena@upo.es [E. Barrena]
jpanaderom@uoc.edu [J. Panadero]; rtorremartinez@gmail.com [R. de la Torre]; ajuanp@gmail.com [A. Juan]

Abstract

Modern transport systems are not only large-scale but also highly dynamic, which makes it difficult to optimize by just employing classical methods. This paper analyzes a realistic and novel problem within the Physical Internet initiative which consists of container transportation throughout a spoke-hub network. Containers need to be transported from their origin locations to their final destinations on or before a given deadline, and they can be temporarily stored in network hubs. Each truck can move one container at a time from one hub to another, containers can be transported by different trucks during their path from their origin to their destination, and drivers need to be back at their starting points in due time. A deterministic heuristic, based on discrete-event simulation, is proposed as a first step to address the intrinsic dynamism of this time-evolving system. Then, in a second step, a biased-randomized version of this heuristic is incorporated into a multi-start framework (BR-MS) to generate better solutions. Next, our methodology is extended to a iterated local search (ILS) framework. Finally, a two-stage algorithm, combining both the BR-MS and the ILS frameworks is proposed. Several computational experiments have been carried out on a set of new benchmark instances, adapted from real road networks, to illustrate the problem and compare the performance of the different solving approaches.

Keywords: heuristics; Biased-randomized heuristics; discrete-event heuristics; large-scale transport networks; dynamic transport systems

1. Introduction

For the last few decades, economic growth and expansion of population have triggered greater demand for road transport. In this environment, companies have to be able to quickly respond to customer needs,

*Author to whom all correspondence should be addressed (e-mail: rtorremartinez@gmail.com).

and thus continuously strive for their supply chain improvement. Hence, logistics and transport systems have become not only global (large-scale) but also increasingly dynamic. However, some forecasting sources claim that this growth has not yet reached its peak. Therefore, for the next decades a drastic increment in road transport is to be expected. In this regard, the weight of shipments moved by road in the United States is anticipated to increase by 26% in 2040 (Chambers et al., 2015). In China, the population of trucks might rise by 21%, up to 14.47 millions in 2030 (Chen and Wang, 2016).

This excessive growth in such a short period of time has contributed to the greenhouse effect, with a sharp increment of energy consumption and pollution, aside from the social impact. For instance, there is a high demand for truck drivers, yet their working hours keep them away from home for long periods of time, which impacts their social life and health. Also, goods often travel unnecessary distances, which can be avoided by linking them smartly and developing interconnected distribution networks. Meanwhile, city logistics activities create significant traffic, noise, and pollution in the community. In order to reduce the aforementioned impact, new innovative logistics approaches are requested. These approaches need to consider both social and environmental aspects. The social dimension is related to compliance with labor regulations, for which the literature offers several approaches. For networks with long-distance transport, the transport costs are determined considering team drivers (Goel, 2018; Wolfinger et al., 2019; Goel et al., 2020). These rotating shifts allow drivers to take mandatory breaks to rest. A second approach is based on a collaborative network, in which drivers are organized by zones, so they can return to their homes at the end of the day (Neves-Moreira et al., 2016).

The integration of sustainability principles in distribution models is also approached from different perspectives. For instance, authors like Neves-Moreira et al. (2016) and Wolfinger and Salazar-González (2021) defend the convenience of using transshipment locations to support a more sustainable distribution system. The transshipment location-allocation problem consists in designing a transport network in which transshipment facilities are located for cargo exchange (e.g., in the form of inter-modal hubs) and to allocate cargo flows through them. In this way, it is allowed to transport the loads from various origins and destinations, in order to satisfy supply limitations, labor restrictions, and demand requirements.

As part of a new logistics paradigm, Montreuil (2011) has developed the Physical Internet (PI) initiative, which shifts towards a more efficient and sustainable global logistics systems: one based on spoke-hub networks and horizontal cooperation among carriers (Serrano-Hernández et al., 2017). Moreover, modern supply chains require greater traceability and transparency. To deal with this issue, Kopetz (2011) presents the concept 'Internet of Things' (IoT), which can be used for connecting physical objects through the World Wide Web. This facilitates the communication between transport systems and their users, thus enabling managers to make informed decisions as the system evolves.

While this approach might speed up deliveries and facilitates short-haul truck driving –which increases the social dimension by allowing drivers to return their homes at the end of each working day–, it is important to consider several trade-offs. Sarraj et al. (2014) compares the current performance of the France road network with the PI-enabled distribution, introducing hubs and implementing multi-modal transportation in order to reduce CO_2 emissions. According to their results, the PI concept could substantially improve logistics efficiency and sustainability, e.g., reducing carbon footprint by 60% without jeopardizing operational costs or accommodating lead times. In the same line, Fazili et al. (2017) confirms that, in comparison with conventional methods, the PI distribution approach reduces driving distance, gas emissions, social burden of traffic, and verifies that the number of drivers who return home on the same day stays comparatively high despite traffic intensity.

In this context, this paper proposes a distribution approach in which a network of hubs allows containers to be moved from their origins to their destinations by a series of cooperative drivers from a collaborative network perspective (i.e., the hub can be the furthest destination for a driver, if that is a labor regulation requirement). Each driver is responsible for moving one container at a time, from one hub to another in the network, until the container reaches its final destination. No predefined paths are given in advance (thus making the routing design become more complex). By doing this, part of the externalities of the transport activity are expected to be palliated, and the sustainable goals achieved: drivers may return home everyday, and the pollutants can be minimized by reaching containers' destinations faster and by reducing the number of empty backhauls.

In such a scenario, and with the main goal of minimizing the total time required to move all containers from their origins to their destinations (while considering social and environmental aspects), several decisions need to be made: *(i)* which container needs to be moved next?; *(ii)* which path must follow the selected container?; *(iii)* which driver is responsible for moving it to the next hub in the network?; and *(iv)* how to deal with the synchronization issues that may arise in this type of dynamic transport systems?. In order to solve this 'dynamic' optimization problem, a novel four-step procedure has been designed:

- First, a fast and simple greedy heuristic is proposed. Similar to the work developed by Fikar et al. (2016), this heuristic is based on discrete-event simulation (DES) concepts: every decision made triggers an event that will modify the state of the system, thus changing future decisions. Like any other deterministic heuristic, the same series of decisions are made for a given instance and, as a consequence, the final solution provided by the greedy heuristic is unique. In particular, the heuristic selects the 'best-next' decision at every step: the most 'urgent' container is selected, the shortest path is chosen, and the closest driver is assigned to move it.
- Secondly, the previous DES-based heuristic is extended into a probabilistic algorithm by incorporating biased-randomization (BR) techniques (Ferone et al., 2019). These techniques make use of skewed probability distributions to introduce some non-uniform randomness into the heuristic constructive process. This way, the deterministic heuristic is transformed into a probabilistic procedure that can be encapsulated into a multi-start (MS) framework (Martí et al., 2013) to generate multiple solutions. Each of these solutions is constructed according to a probabilistic version of the criteria used by the deterministic heuristic. BR techniques have been successfully employed to solve different combinatorial optimization problems in multiple application fields: vehicle routing (Dominguez et al., 2014; Quintero-Araujo et al., 2017), facility location (Pagès-Bernaus et al., 2019), scheduling (Ferrer et al., 2016), etc.
- Thirdly, the MS framework is extended into a iterated local search (ILS) metaheuristic framework (Lourenço et al., 2010), which does not require to re-construct an entire solution from scratch.
- Finally, a two-stage algorithm, combining both the MS and ILS frameworks, is proposed. In the first stage, the MS allows for 'diversification', and produces a reduced pool of promising solutions. In the second stage, the ILS framework allows for 'intensification', exploring in more detail each of the promising solutions in the pool.

Accordingly, the main contributions of this paper are: *(i)* the description of a coordinated multi-vehicle routing problem in interconnected networks; *(ii)* the design, implementation, and testing of different heuristic-based solving approaches, including a novel type of optimization heuristic based on concepts

from discrete-event simulation; and *(iii)* the generation of a set of benchmarks that can be used for testing other solution approaches to this problem. The remaining of the paper is organized as follows: Section 2 provides a review of the literature on the Physical Internet concept and on container distribution models; Section 3 provides more details on the coordinated multi-vehicle routing problem analyzed in this paper; Section 4 presents a Mixed Integer-Linear Programming (MILP) Model that is used to solve small-sized instances, which contributes to validate the proposed heuristics; the building of the solving approach is explained in Section 5; Section 6 presents an extensive set of computational experiments and explains how the instances were created; Section 7 extends the analysis of the results; finally, Section 8 outlines the main conclusions and provides some suggestions for future research.

2. Related Literature

This section reviews previous contributions related to the idea of interconnected logistics networks (supported by the PI concept and by the use of IoT solutions) as well as to existing work on container-distribution systems.

2.1. The Physical Internet

Unsustainability in the way physical objects are transported, handled, stored, realized, supplied and used, is well known within the academia (Montreuil, 2011). For instance, long periods of time in which drivers have to be away from home have generated a shortage of drivers due to low job satisfaction. Considering the future growth in the transportation field, this can end into a bottleneck that may limit future development (Costello and Suarez, 2015). Hence, the proposed optimization model aims to solve the problem by introducing 'socially responsible' schedules that allow drives to make their jobs compatible with their personal lifestyles.

In this regard, the PI makes it possible to improve efficiency and sustainability (in its three dimensions) through experiments based on analytical, optimization, and simulation models. Montreuil (2011) proposed to strive towards the PI, where logistics networks would be interconnected and goods would be encapsulated in world-standardized, green, networked, and smart containers that can be distributed across fast, reliable, and eco-friendly transportation systems. A wide review on the PI concept has been gathered by Pan et al. (2017). In a similar line, Treiblmaier et al. (2020) provides a literature review on the PI concept, as well as an analysis on its future directions. Likewise, Sallez et al. (2016) review research papers on smart containers, and explain various projects which aim to achieve different levels of 'container intelligence' in their communication with supply chain management systems by using IoT solutions. Gubbi et al. (2013) provides the concept, the architectural elements, and the future developments, whereas Díaz et al. (2016) tackles challenges of IoT implementation, data storing, analytics, and security. Some of the most relevant real-world experiences are described next. Hakimi et al. (2015) focus on the interconnected transport of semi-trailers, which use relay-based exchanges at PI transit centers in Quebec. Meller et al. (2012) design an assessment based on an analytical model of the US supply chain. Sohrabi et al. (2016) present an optimization model for these type of systems.

Considering the main characteristics of our problem, dynamic traffic information will affect freight

movement, allow better planning, and improved scheduling (Gubbi et al., 2013). Online monitoring of origin-destination routes and container travel times will enable us to make efficient decisions depending on the state of the system, e.g.: to compute routes and assign available drivers to containers according to priority, as the containers enter or progress within the distribution network.

2.2. The Container Distribution Model

Up to now, a tremendous attention has been given to optimization of diverse logistics networks. There is a wide range of literature available on optimization in different areas, such as road logistics, marine networks, or cargo industry in general. Just to point out some recent examples, Funke and Kopfer (2016) builds a model for a multi-size inland container transport problem, which aims to minimize the total travel distance and operation time of the trucks. He et al. (2015) offers an optimization model for a multi-echelon container supply chain, whereas Hao and Yue (2016) optimizes a multi-modal transport system by combining routes.

In the context of hub-and-spoke systems, Ji and Chu (2012) optimize a port logistics network of dynamic hinterland, while Ji et al. (2015) study routing optimization for multi-type containerships with time deadline to minimize service, waiting, and traveling cost. Wang (2008) combines two hub-and-spoke networks into a regional port cluster. Zäpfel and Wasner (2002) concentrate on decision-making, calculating cost savings associated with developing a basic hub-and-spoke or a hybrid version, and applying the model to the Australian parcel service provider. Along with routing decisions, Hsu and Hsieh (2007) also consider inventory levels, with the goal of minimizing shipping and inventory costs.

Quite often, combinatorial problems in transport and logistics are particular cases and variations of the network design problem (Magnanti and Wong, 1984), where some sort of discrete choice is involved. Typically, this choice relates to the network structure. One related topic is the multi-commodity network design (MCND) problem (Gendron et al., 1999). It has many applications in transport planning whenever certain commodity demands are to be satisfied, but also decisions can be made about the location of several facilities. Here, containers can be understood as integer commodities that need to be routed with the specific origin and destination nodes. This scenario was introduced by Montreuil (2011) in order to switch from the traditional point-to-point delivery scenario (in which drivers complete a multi-day journey and where long-distance backhauls are frequent) to a PI-enabled model, where the driver brings a container from one hub to another that can be reached in a few hours, and then returns home –possible carrying another container that has to be moved in the opposite direction.

Some of the aforementioned approaches share similarities (as some of them are closely related to the classical pickup and delivery problem). In particular, the problem analyzed in Neves-Moreira et al. (2016) is similar to the one we propose. Still, our work differs from theirs in at least the following aspects: *(i)* our model assumes that each container has to be transported to its destination, otherwise the solution is not a feasible one; *(ii)* the objective function in our approach consists in minimizing the total time employed since the first driver is activated until the last driver returns home, while their objective function simultaneously maximizes the movements performed by each request and minimizes the time that is necessary to execute the distribution plan; and *(iii)* we propose a novel solving approach that is inspired in iteratively processing a list of dynamic events scheduled at a discrete number of times. This approach, inspired in a deterministic discrete-event simulation, can provide reasonably good solutions

in real-time by parallelizing the biased-randomized heuristic. In addition, by considering a stochastic discrete-event simulation, it can easily be extended to a simulation-optimization algorithm (Chica et al., 2020), so it can deal with stochastic travel times. Our approach is also flexible in terms of routing optimization, as there are no driver-container pairs that need to start and return together, as in Fazili et al. (2017). Thus, paths of containers and drivers are rather calculated independently.

3. A More Detailed Description of the Transport Network Problem

The spoke-hub network problem studied can be defined by means of a graph $N = (L \cup H \cup D, E)$, where L is the set of nodes representing origins and destinations of each container, H is the set of hub nodes in the network (locations where a container can be temporarily stored), D is the set of depots where drivers are initially available, and E is the set of edges connecting some of the previous nodes (i.e., the graph does not have to be complete). Each edge $e = \{i, j\} \in E$ is characterized by a travel time required to traverse it, $\tau_{ij} > 0$. We consider a set $\mathcal{R}$ of drivers. Each driver $d \in \mathcal{R}$ has an associated home depot $h_d \in D$, which is the starting and ending node of the corresponding driver route. Depot nodes $h_d \in D$ cannot store containers, neither be visited by a driver with a different home depot node. Also, a set of containers C has to be transported: each container $c \in C$ has an associated origin and destination ($o_c, d_c \in L$), and a due time ($\tau_c > 0$) by which the container has to be at its destination. Containers can be dropped temporarily at hubs on their way from their origin to their destination. We also consider a planning horizon T.

It is assumed that hubs are virtually uncapacitated (i.e., they can store any number of containers). The goal is to minimize the total time required to complete the distribution process, which includes the delivery of all containers and the return of all drivers to their associated origins. The following constraints apply: *(i)* each truck can load and transport one container at a time; *(ii)* all containers must reach their destination on or before the corresponding due time; *(iii)* there is a maximum number of working hours per driver (w_d), after which the driver is unavailable until the next day ; and *(iv)* depending on the number of drivers and the needs of the system shipments, some drivers may be required only once (by positioning the driver in advance to the transfer hub such, so the trip can be continued immediately). Figures 1 and 2 show, respectively, a representation of a small network and a feasible solution for a problem with 2 containers and 3 drivers, where the symbol 'endpoint' represents the origin of some containers and the destination of others. In this example, nodes 4 and 5 are the origins of containers 1 and 2, respectively. Likewise, nodes 6 and 7 are the respective destination of each container. Figure 2 only reports the nodes where drivers stop, and not all the nodes they traverse in their route.

4. A Mixed Integer-Linear Programming Model

The formulation of the problem is defined over the directed graph $(L \cup H \cup D, A)$, which is the result of converting the undirected graph N described Section 3 into a directed graph. We use binary variables x_{ij}^{dc}, which take the value 1 if, and only if, arc $(i, j) \in A$ is used by driver $d \in \mathcal{R}$ to transport container $c \in C \cup \{0\}$, being container 0 a virtual one used to represent empty movements. We also consider the following additional variables:

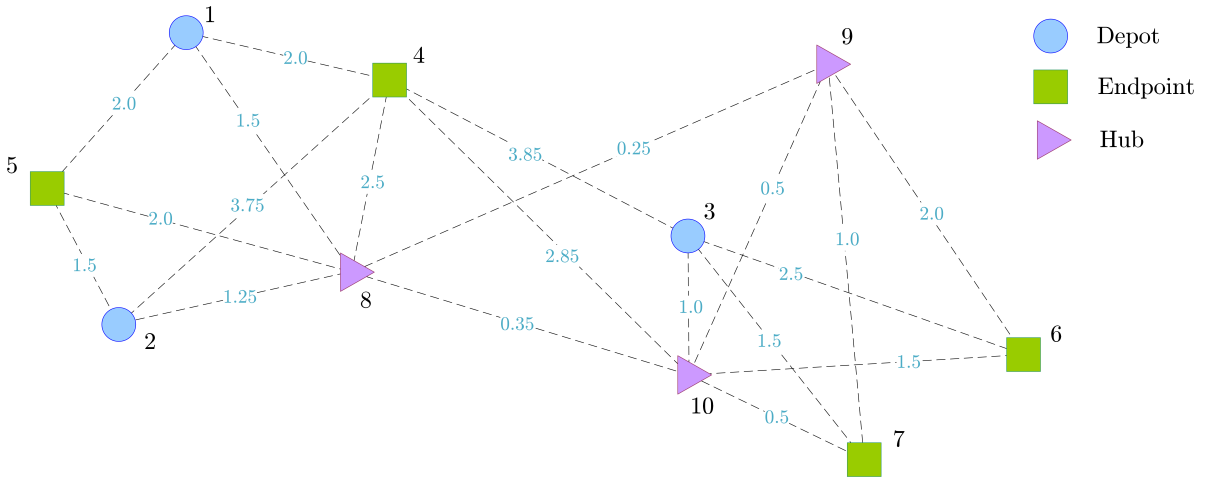

Fig. 1: A simple example of the coordinated multi-vehicle transport problem.

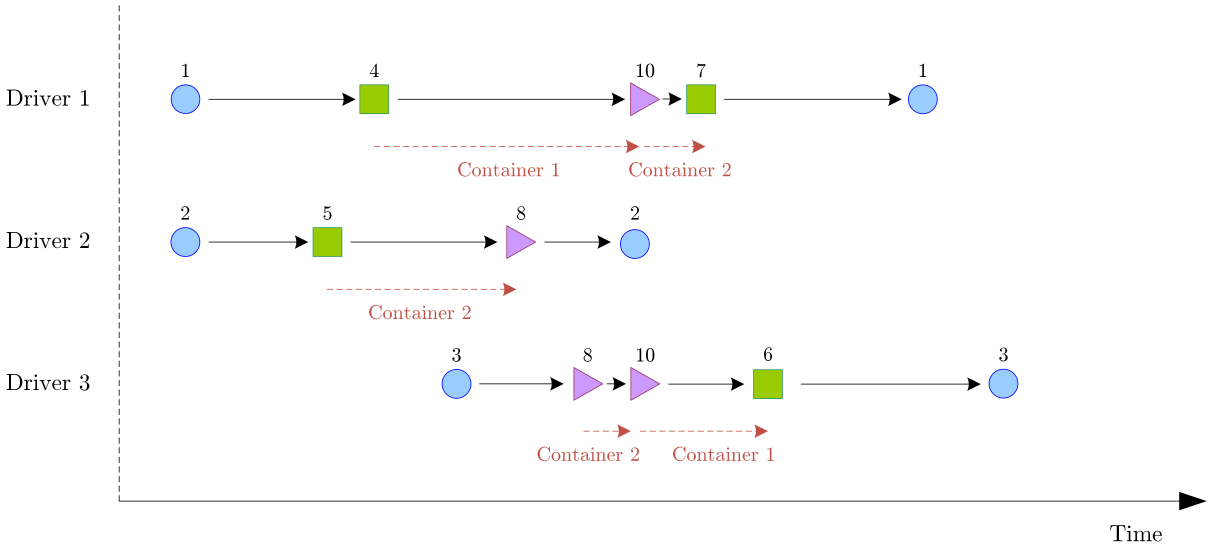

Fig. 2: Representation of a feasible solution.

- U: threshold variable, $U \in [0, T]$, representing an upper bound on the return time of drivers to the home depot (this variable will be minimized in the objective function).
- t_i^d: arrival time of driver d at node $i \in N$.
- t_0^d: departure time of driver d from the associated home depot h_d.
- $\hat{t}_i^c$: arrival time of container c at node $i \in L \cup H$.

The main sets and parameters are summarized in Table 1. The basic version of the problem assumes

Table 1: Summary of the main sets and parameters

SETS & PARAMETERS	DESCRIPTION
$N = (L \cup H \cup D, E)$	initial graph
L	the set of nodes representing origins and destinations of each container
H	the set of hub nodes in the network
D	the set of 'depots' where drivers are initially available
E	the set of edges connecting some of the previous nodes
$\mathcal{R}$	set of drivers
C	set of containers that have to be transported
τ_{ij}	travel time required to traverse edge $e = \{i, j\} \in E$
T	planning horizon
$h_d \in D$	associated 'home' depot for driver $d \in \mathcal{R}$
w_d	maximum number of working hours for driver $d \in \mathcal{R}$
$o_c \in L$	origin node of container $c \in C$
$d_c \in L$	destination node of container $c \in C$
τ_c	due time by which the container $c \in C$ has to reach its destination

that each driver can visit a node at most once. This version can be formulated as follows:

$$\text{Minimize} \quad U \tag{1}$$

subject to Constraints (3) to (16), which are described in the following sections.

4.1. Global Time Threshold Constraints

Constraints (2), in combination with the objective function, minimize the last arrival time of drivers to their associated home depot at the end of the trip (we assume that starting time of the distribution process for all drivers is 0):

$$t_i^d \leq U \qquad \forall d \in \mathcal{R}, \quad \forall i \in N \tag{2}$$

Notice that if we do not assume that all drivers will start at time zero, then the left part of Constraints (2) would be $t_i^d - t_0^d$. Constraints (3) impose that the arrival time of any driver $d \in \mathcal{R}$ at any node $i \in N$ must be less or equal that T (planning horizon):

$$t_i^d \leq T \qquad \forall d \in \mathcal{R} \quad \forall i \in N \tag{3}$$

Notice that constraints (3) are redundant with Constraints (2). However, we keep both in order to strengthen the formulation.

4.2. Departure and Arrival Constraints

Constraints (4) ensure that each container is transported (either by one or multiple drivers) from its origin, and it arrives to its destination:

$$\sum_{d\in\mathcal{R}}\sum_{j\in(N\setminus D)\cap\delta^+(o_c)} x^{dc}_{o_c j} = \sum_{d\in\mathcal{R}}\sum_{j\in(N\setminus D)\cap\delta^-(d_c)} x^{dc}_{j d_c} = 1 \qquad \forall c\in C \tag{4}$$

Note that $\delta^+(i)$ and $\delta^-(i)$ are the sets of successors and predecessors, respectively, of node $i \in N$.

Constraints (5) ensure that each driver who starts a trip must finish it. Notice that the starting and ending node for a driver d is home depot h_d. We also assume that each driver makes at most one trip, even though several containers may be transported during the trip (one at a time):

$$\sum_{c\in C\cup\{0\}}\sum_{j\in\delta^+(h_d)} x^{dc}_{h_d j} = \sum_{c\in C\cup\{0\}}\sum_{j\in\delta^-(h_d)} x^{dc}_{j h_d} \le 1 \qquad \forall d\in\mathcal{R} \tag{5}$$

4.3. Flow Conservation Constraints

Constraints (6) ensure that, at each non-depot node $j \in H \cup L$, if a container arrives to it then the container must also leave it (unless the node is the origin or destination of the container). Notice that we use the less-or-equal symbol, since each container is transported by at most one driver at a time:

$$\sum_{d\in\mathcal{R}}\sum_{i\in\delta^-(j)} x^{dc}_{ij} = \sum_{d\in\mathcal{R}}\sum_{i\in\delta^+(j)} x^{dc}_{ji} \le 1 \qquad \forall c\in C \quad \forall j\in H\cup L, \quad j\neq o_c, d_c \tag{6}$$

Constraints (7) ensure that, at each non-depot node $j \in H \cup L$, if a driver arrives to it then the driver must also leave it. Again, we use the less-or-equal symbol (since each driver can transport at most one container at a time):

$$\sum_{c\in C\cup\{0\}}\sum_{i\in\delta^-(i)} x^{dc}_{ij} = \sum_{c\in C\cup\{0\}}\sum_{i\in\delta^+(i)} x^{dc}_{ji} \le 1 \qquad \forall d\in\mathcal{R} \quad \forall j\in H\cup L \tag{7}$$

4.4. Time and Subtour Elimination Constraints

Constraints (8) impose that, if a driver traverses the arc (h_d, j), the arrival time at node j must be at least the departure time from the home depot h_d (t^d_0) plus the time required to traverse the arc (h_d, j):

$$t^d_0 + \tau_{h_d j} \le t^d_j + M\Big(1 - \sum_{c\in C\cup\{0\}} x^{dc}_{h_d j}\Big) \qquad \forall d\in\mathcal{R} \quad \forall j\in\delta^+(d) \tag{8}$$

Here, we make use of a 'big enough' constant M, e.g.: $M = 2T$. Constraints (9) impose that if a driver traverses arc (i, j), the arrival time at node j must be at least the departure time from node i plus the time required to traverse arc (i, j):

$$t_i^d + \tau_{ij} \leq t_j^d + M(1 - \sum_{c \in C \cup \{0\}} x_{ij}^{dc}) \qquad \forall d \in \mathcal{R} \quad \forall j \in \delta^+(i), \quad \forall i \in H \cup L \tag{9}$$

for a big enough M, e.g., $M = 2T$. Constraints (10) and (11) define the arrival time of container $c \in C$ at node $j \in H \cup L$. If a driver transports a container, the arrival time of the container equals the arrival time of the driver who transports it:

$$\hat{t}_j^c \leq M \sum_{i \in \delta^-(j)} \sum_{d \in \mathcal{R}} x_{ij}^{dc} \qquad \forall c \in C \quad \forall j \in H \cup L \tag{10}$$

$$t_j^d - M(1 - \sum_{i \in \delta^-(j)} x_{ij}^{dc}) \leq \hat{t}_j^c \leq t_j^d + M(1 - \sum_{i \in \delta^-(j)} x_{ij}^{dc}) \quad \forall c \in C \quad \forall d \in \mathcal{R} \quad \forall j \in H \cup L \tag{11}$$

Constraints (12) impose that if a container traverses arc (i, j), the arrival time at node j must be at least the departure time from node i plus the time required to traverse the arc (i, j):

$$\hat{t}_i^c + \tau_{ij} \leq \hat{t}_j^c + M(1 - \sum_{d \in \mathcal{R}} x_{ij}^{dc}) \qquad \forall c \in C \quad \forall j \in \delta^+(i), \quad \forall i \in H \cup L \tag{12}$$

Once more, we consider a 'big enough' constant M, e.g., $M = 2T$. Constraints (13) ensure that all containers must reach their destination on or before the corresponding due time:

$$\hat{t}_{d_c}^c \leq \tau_c \qquad \forall c \in C \tag{13}$$

Constraints (14) ensure that the maximum number w_d of working hours per driver $d \in \mathcal{R}$ and day is not exceeded:

$$t_{h_d}^d \leq w_d \qquad \forall d \in \mathcal{R} \tag{14}$$

4.5. Variables that can be Set to Zero or not Defined

As exposed in Section 3, the depots in D cannot store containers, and each of them acts as the home base for the drivers associated with it. Constraints (15) indicate that drivers cannot depart from home depots with a container (i.e., outgoing arc at drivers' home depots are empty movements):

$$x_{ij}^{dc} = 0 \qquad \forall d \in \mathcal{R}, \quad \forall c \in C, \quad \forall i \in D, \quad \forall j \in \delta^+(i) \tag{15}$$

Constraints (16) indicate that a driver cannot use an outgoing empty arc from another driver's home depot:

$$x_{ij}^{d0} = 0 \qquad \forall d \in \mathcal{R}, \quad \forall i \in D \setminus \{h_d\}, \quad \forall j \in \delta^+(i) \tag{16}$$

As described in Section 3 and forced by Constraints (15), drivers depart and arrive from/to their home depot without containers. Hence, Constraints (5) can be replaced by Constraints (17):

$$\sum_{j \in \delta^+(h_d)} x_{h_d j}^{d0} = \sum_{j \in \delta^-(h_d)} x_{j h_d}^{d0} \leq 1 \qquad \forall d \in \mathcal{R} \tag{17}$$

Notice that the formulation of the problem can also be defined over the complete directed graph $(L \cup H \cup D, A^*)$, where the travel time τ_{ij} between each pair of nodes is computed as the shortest path over the initial graph $(L \cup H \cup D, A)$. In this way, nodes that lie on a shortest path may also be intrinsically visited more than once. Since the complete graph considerably increases the size of the problem –and, therefore, the number of variables–, we apply a pre-processing phase and a valid inequality to reduce this size. As indicated in El-Hajj et al. (2016), one way of reducing the size of a problem is by considering just accessible nodes and arcs. More specifically, we test two main types of pre-processing by disregarding, for each driver, the nodes and arcs that cannot be visited during the w_d time threshold. We also make use of a valid inequality, on the global duration of all routes, proposed in Bianchessi et al. (2018).

5. Solving Methodology

This section describes four incremental approaches to solve the aforementioned problem. First, a greedy heuristic based on concepts from discrete-event simulation is presented. This DES-based heuristic makes deterministic decisions based on the 'best-in-the-short-term' selections related to containers, drivers, and paths. Later, a biased-randomized multi-start version of the previous heuristic is proposed. Then, a ILS metaheuristic is introduced to explore intermediate states of solutions. Finally, a 2-stage algorithm combining the multi-start and the ILS frameworks is designed to widely explore the search space in a more efficient manner.

5.1. DES-based Heuristic

In discrete-event simulation, every time an event occurs, and a new decision is made to deal with it, new events might be triggered and processed as the 'simulation clock' evolves over time (White and Ingalls, 2015). These new events are then scheduled to happen in a future time, which might have a deterministic

or a random nature. Hence, every decision made can trigger a future event, which might change the state of the system once it occurs. In our case, there are three types of events:

- *Activation of a driver for a possible pickup:* a 'pickup' event consists in moving a free driver from his / her current location towards a node where a container is likely to require transport.
- *Transport of a container (transport):* a 'transport' event consists in moving a container from its current location to a new node in the network –this new node could be either its destination or an intermediate hub, and reaching it might require traversing other intermediate nodes.
- *Returning a driver to the associated home depot:* a 'return' event consist in requesting the driver to return to the home depot, so that the working shift is not exceeded –the driver can still transport containers during the return to the depot as far as that is compatible with being at home on time.

From these possible events, the following DES-based heuristic is proposed with the goal of minimizing the total time required to transport all containers and return all drivers to their home depot:

1. Compute the shortest path between any pair of nodes in the network using the Floyd-Warshall algorithm (Floyd, 1962).
2. Assign drivers to containers in order to mobilize as many containers as possible. While a container does not reach the final destination, it is considered a 'travelling' container. In case there are more containers than drivers, prioritize containers according to their deadlines and distance from their destination. In other words: the most 'urgent' containers are assigned to their nearest drivers until no more drivers are available, or all containers have been assigned. This initial list of pickup events will generate future transport events that will be scheduled according to the deterministic times required to traverse the necessary edges. In turn, as new decisions are made on these transport events, new pickup, transport, or return events will be added to the list and scheduled to happen in a future time.
3. Steps 4 and 5 are iteratively executed until there are no more events in the list –i.e., until all containers have been delivered, and all drivers have returned to their depots.
4. In each iteration, the next scheduled event is removed from the list and processed, while the (deterministic) simulation clock is updated to that time. Processing an event means making decisions about the next steps (e.g., which is the next container to be moved?, which driver should be activated?, which is the next node to bring the container to?, etc.). Accordingly, these decisions will trigger new events that need to be added to the list and be scheduled in the future.
5. After each new event, update the state variables and statistics of the system (e.g., current locations of drivers and containers, etc.).
6. Once all events in the list have been processed, a solution has been generated and its time-based cost can be computed. Based on the defined goal, this cost is given by the time in which the last driver returns to the associated home depot after all containers have reached their destination nodes.

5.2. Extending the DES-based Heuristic to a Biased-Randomized Multi-Start Procedure

As with any other heuristic, the DES-based one follows a greedy behaviour, i.e.: the best choice in the short term is always chosen. So, the solution generated by this deterministic procedure is always the same regardless if it is executed multiple times.

In order to overcome this disadvantage and to explore a wider search area, a biased-randomized multi-

start (BR-MS) version of this heuristic is designed. As discussed in Belloso et al. (2019), this can be achieved by using skewed probability distributions instead of a greedy behaviour while constructing the solution. In our case, the selections of the next container, next path (next node to visit), and next driver are randomized by using Geometric probability distributions with parameters β_c, β_p, and β_d (respectively), with each $\beta \in (0, 1)$. The use of the Geometric probability distribution is justified by the fact that it only contains one parameter, β, which is easy to set (Grasas et al., 2017). In addition, values of β closer to 1 induce a more greedy behavior in the algorithm, while values of β closer to 0 induce a more uniformly random behavior. Notice that are the values in between the most interesting ones, since they correspond to a behavior between greedy and uniformly random, which has been successfully tested in many routing and scheduling applications (Dominguez et al., 2016; Ferone et al., 2020). By introducing this non-uniform randomization effect, a new solution is likely to be generated every time the procedure is run. Still, because of the skewed probability distribution employed, the logic behind the heuristic is respected, i.e., the new selection behaviour is somewhat between greedy and random uniform.

The scheme of the algorithm is a natural extension of the one for the heuristic. However, the following variations occur as shown in Algorithm 1: *(i)* not always the most 'urgent' container is selected next –still, one of the most urgent ones is likely to be selected; *(ii)* not always the shortest path towards the container's destination is selected –still, one of the shortest path will be selected; and *(iii)* not always the driver who is closer to the current location of the container will be assigned to it –although one of the drivers in the close neighborhood is likely to be selected.

Thus, the path that each container will follow is not set in the beginning. On the contrary, it is decided at every step of the algorithm. In order to select the next node (a hub or the container's destination) the following steps are followed:

1. Given the location of a container, generate a list of the adjacent hubs.
2. Sort this list by the time requested to travel from each intermediate hub to the container's destination (following the shortest path) and taking into account the time required to travel the edge connecting the current location of the container with the intermediate hub.
3. Use the Geometric probability distribution to randomly select the next hub from the sorted list.

The idea behind this approach is that the best option in the short run does not necessarily lead to the best global solution (in some cases, not even to a feasible one).

5.3. Extending the DES-based Heuristic to a Biased-Randomized Iterated Local Search

The BR-MS approach is useful to explore different routing possibilities during the search for high-quality solutions. Nonetheless, this search starts from scratch at each iteration without taking advantage of the existence of already good solutions. This section describes how to extend the DES-based heuristic into a biased-randomized iterated local search (BR-ILS), which makes uses of a 'base' solution and a destruction-construction process to avoid starting new solutions from scratch. Thus, while the BR-MS has a higher 'exploration' component (it explores better the different regions of the solution space), the BR-ILS has a higher 'exploitation' component (it intensifies the search around good solutions already found).

In our case, a partial solution x_p is the result of undoing the last $p\%$ of decisions taken in a com-

Algorithm 1 The Biased-Randomized Discrete-Event Heuristic (BRH)

```
1: procedure ScheduleContainerEventsBRH(β_c, β_p, β_d, containers, drivers, solution, shortestPaths)
2:     travellingContainers ← getTravellingContainers(containers)
3:     if travellingContainers is not empty then
4:         sort(travellingContainers)                                   ▷ by urgency
5:         idleDrivers ← getIdleDrivers(drivers)
6:         k ← 0
7:         while k < size of travellingContainers and idleDrivers is not empty do
8:             container ← selectContainerBRH(β_c, travellingContainers)
9:             node ← selectPathBRH(β_p, container, shortestPaths)
10:            driversList ← sort(idleDrivers)                          ▷ by time-based cost
11:            found ← false
12:            while driversList is not empty and not found do
13:                driver ← selectDriverBRH(β_d, driversList)
14:                if areFeasible(driver, container) then
15:                    scheduleEvent(driver, container, node)
16:                    updateSolution(solution)
17:                    removeDriver(idleDrivers)
18:                    removeContainer(travellingContainers)
19:                    found ← true
20:                else
21:                    removeDriver(driversList)
22:                end if
23:            end while
24:            k ← k + 1
25:        end while
26:    end if
27: end procedure
```

plete solution x. Therefore, the 'perturbation' phase in our ILS consists in: *(i)* partially *destroying* a base solution x (the size of the destruction is given by the varying parameter p); and then *(ii)* completing the partial solution by *reconstructing* it according to the biased-randomized version of the DES-based heuristic. Once a new complete solution has been obtained, if it outperforms the base solution or the best solutions, these are updated and the percentage of destruction p is reset to its minimum value p_{min}. Otherwise, p is iteratively incremented by a certain $step$ until it reaches its maximum value p_{max}. Notice that, occasionally, in order to further diversify the search, the algorithm, following an acceptance criterion, might accept non-improving solutions. This criterion is based on the distance between the costs of the new and the base solution ($demon$ value). This mechanism helps to prevent getting stuck in local minimums. This procedure is summarized in Algorithm 2.

5.4. A Two-Stage hybrid Algorithm Combining BR-MS with BR-ILS

The BR-ILS is an effective way to intensify the search in those promising regions of the solution space where the initial solution was found. Notice, however, that the time-dependant nature of the events implies that every decision will change the future evolution of the network (drivers' availability, containers' location, etc.). Hence, the BR-ILS approach could invest too much time in partially destroying and recon-

Algorithm 2 The BR-ILS Algorithm

```
1: procedure SolveBRILS(network, containers, drivers, iter_max, p_min, p_max, step)
2:     % p_min: minimum percentage of solution destruction
3:     % p_max: maximum percentage of solution destruction
4:     % step: increment of percentage on each iteration

5:     baseSol ← SolveSH(network, containers, drivers)          ▷ Using the DES-based heuristic
6:     demon ← 0
7:     bestSol ← baseSol
8:     p ← p_min
9:     iter ← 0
10:    while iter < iter_max do
11:        newSol ← perturbation(p, baseSol)                    ▷ Perturbation phase
12:        Δ ← totalTime(newSol) − totalTime(baseSol)
13:        if Δ < 0 then                                        ▷ Improvement
14:            baseSol ← newSol
15:            p ← p_min − step
16:            demon ← −Δ
17:            if totalTime(newSol) < totalTime(bestSol) then
18:                bestSol ← newSol
19:            end if
20:        else
21:            if Δ < demon then                                ▷ Worsening
22:                baseSol ← newSol
23:                p ← p_min − step
24:                demon ← 0
25:            end if
26:        end if
27:        p ← min(p + step, p_max)
28:        iter ← iter + 1
29:    end while
30:    return bestSol
31: end procedure
```

structing base solutions which initial decisions (the ones corresponding to the first events) compromise the quality of the final solution. In other words, the existence of these inter-dependencies between past and future events limits the efficiency of local search operators (since random movements might compromise the feasibility of time-synchronized decisions). Without an effective local search procedure, the BR-ILS might get easily trapped into a local minimum unless the base solution is completely reset from time to time.

Thus, in order to avoid investing too much computing time intensifying the search around a sub-optimal base solution, we propose a 2-stage hybrid algorithm (HILS) as follows: in the first stage, the BR-MS procedure is employed to widely explore the solution space and identify a reduced set of very promising solutions. Then, in the second stage, each of these promising solutions is employed as a starting base solution in the BR-ILS procedure.

6. Computational Experiments

This section aims to provide a structured insight into the following contents: *(i)* a short description of the experiments over a reduced set of three small instances for both the deterministic DES-based heuristic and an exact approach for solving the mixed integer-linear programming (MILP) model described in Section 4; *(ii)* a characterization of the adapted networks; *(iii)* an extensive description of the procedural strategy to obtain the benchmark instances; *(iv)* a summary of the main characteristics of the 20 benchmark instances created (including the graphical representation of an instance); *(v)* the setting of the algorithm's parameters, the programming language, and the characteristics of the computer employed; and *(vi)* a description of the computational experiments performed as well as their corresponding results.

All algorithms have been implemented using Java SE 8.0_151, and tested with JUnit 4.12. All the experiments were run in a Dell Workstation Precision Tower Serie 7000, Intel Xeon E5-2650 v4 with 32GB RAM In order to assess the quality of the solutions provided by the proposed algorithm, we have carried out a comparison with the MILP exact model presented in Section 4, which has been solved using CPLEX. As only small instances of the MILP model can be solved in practice (i.e., without running out of memory or investing an unreasonable amount of computing time), we have generated a set of four small instances with networks of up to 18 nodes and 2-3 containers. Figure 3 illustrates a graphical representation of the underlying network for instance pi-small-03, which contains 18 nodes and 3 containers. Table 2 presents the obtained results for both our algorithm and the exact solver, where *TT* refers to the total time necessary to deliver all containers and return all drivers to their depots. Notice that the proposed algorithm is able to reach the optimal value provided by the exact solver for these four instances, which contributes to validate its quality. Regarding computational times, the proposed algorithm is able to reach the optimal values in about 0.02 seconds (on the average), whilst the exact solver needs about 35 seconds.

Table 2: Small instances characteristics and results for our algorithm and an exact solver.

	Network			Problem		Our Algorithm		CPLEX		Gaps
Instance	Nodes	Edges	D/H/E	#C	#D	TT [1]	Comp. Time (s) [2]	TT [3]	Comp. Time (s) [4]	[1-3] (%)
pi-small-01	10	20	3/3/4	2	2	14.85	0.02	14.85	0.61	0
pi-small-02	8	13	2/2/4	2	2	6.50	0.02	6.50	0.12	0
pi-small-03	18	51	3/9/6	3	3	6.50	0.03	6.50	138.61	0
pi-small-04	13	78	3/6/4	2	3	5.80	0.02	5.80	1.56	0
					Avg.	8.41	0.023	8.41	35.23	0

Due to the novelty of the considered problem, there are not standardized benchmark instances in the literature. Hence, in order to generate the benchmark instances we decided to adapt the large-scale networks introduced in Keenan (2017) for the time capacitated arc routing problem (De Armas et al., 2019). These datasets are included in the *TCARP large rural datasets*, in which a simplified version of some Irish roads are drawn. Table 3 shows the main characteristics of each instance, including: the number of nodes, the number of edges, the associated density (or sparsity) $d \in (0, 1)$, and basic statistics on the weights (travel times) and the degree of the nodes.

For each network, 4 different instances have been created, each with a distinctive set of containers,

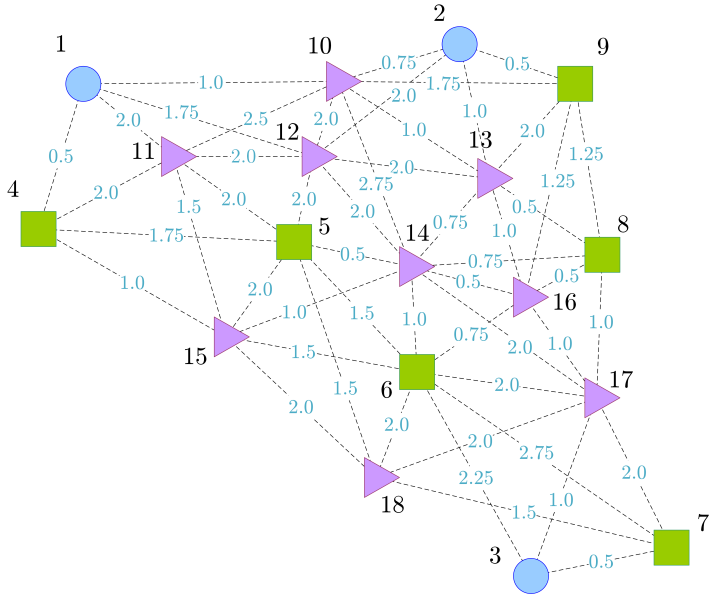

Fig. 3: Representation of the pi-small-03 instance.

Table 3: Basic details of each considered network.

	Network			Weights			Degree	
Instance	Nodes	Edges	Density	Mean	Min	Max	Mean	Max
TCARP-R1	221	245	0.010	0.870	0.069	1.786	2.22	5
TCARP-R2	382	424	0.006	0.938	0.048	2.125	2.22	4
TCARP-R3	508	550	0.004	0.916	0.062	2.893	2.17	5
TCARP-R4	805	872	0.003	0.788	0.022	3.098	2.17	6
TCARP-R5	599	647	0.004	0.964	0.002	3.062	2.16	5

fleet of drivers, and network structure. Specifically, we have considered scenarios with 25, 50, 100, and 200 containers with different upper and lower bounds for the number of drivers per depot as well as for the shifts. The containers have different origins, destinations, and deadlines. Also, each instance has a different number of depots, hubs, and endpoints (origins or destinations). Next, some additional details are provided on how these instances have been created from the original ones:

- *Classification of nodes in the network:* each node in the network has been labeled as a depot, hub, or endpoint according to its connectivity degree, d:
  1. If $d = 1$, it can either be an endpoint with probability 0.5 or a depot with the same probability.
  2. If the node has relatively high connectivity ($d \geq 4$), it is labeled as a hub with probability 0.75 or a depot with probability 0.25.

3. If the degree is $2 \leq d \leq 3$, the node is labeled as a hub with probability 0.5, as a depot with probability 0.3, or as an endpoint with probability 0.2.

- *Configuration of the set of containers to be delivered:* origin and destination endpoints are randomly chosen, but taking into account that the shortest path between both nodes cannot exceed 24 hours for instances 1 to 10 and 12 hours for instances 11 to 20. The due times for the containers are set proportional to the length of the shortest path from their origin to their destination endpoints. Therefore, if the length of the shortest path is l hours, a margin of $k \cdot l$ hours is set, where k usually takes a value between 2 and 3, depending on the instance.
- *Assignment of drivers to depots and configuration of shifts:* the number of drivers assigned to each depot is randomly generated within a lower and upper bound. The working shifts range from 8 to 10 hours.

For each of the 20 benchmarks, Table 4 shows: the size of the network, the number of depots, hubs, and endpoints, the number of containers and drivers, and the parameters used to create each instance. These parameters include: the minimum and the maximum number of drivers per depot, due times per container, and length (in hours) of the shortest path between the origin and the destination. It also displays a summary of the container shipments, including: the maximum, minimum, and average length.

Figure 4 shows the structure of the network for the instance *pi-02*, where circle nodes represent depots, triangular nodes represent hubs, and square nodes represent endpoints.

Regarding the parameters' configuration, the following ranges have been defined for randomly selecting the beta values associated with each of the Geometric distributions: $\beta_c \in (0.70,\ 1.00)$, $\beta_p \in (0.98,\ 1.00)$, and $\beta_d \in (0.85,\ 1.00)$. The aforementioned ranges have been obtained after testing different values of β for the entire set of instances. As explained in Juan et al. (2013, 2014), our aim is to develop algorithms that are relatively simple to implement and use a reduced number of parameters with few setting requirements, which facilitates the associated fine-tuning process. Hence, our algorithm should perform well for any instance and data as far as the β parameters take values inside 'reasonable' intervals. For the three solving approaches (BR-MS, BR-ILS, and HILS), the maximum computational time was set to 180 seconds. Again, after a quick trial-and-error test, the configuration of parameters for the BR-ILS are: $p_{min} = 50$, $p_{max} = 100$, and $step = 2$. In the case of the HILS, the *diversify* parameter was set to 0.5 (i.e., half of the time the hybrid algorithm is executing the BR-MS component and half of the time is executing the BR-ILS one). Finally, the cooldown parameter was set to 10.

Additionally, in order to assess the impact –in terms of the quality of the solution obtained– of a scenario in which drivers' shifts are limited to 8-10h (instead of more typical schedules of 24 / 48h, with the required stops to rest and sleep), a set of computational experiments has been carried out for instances pi-01 to pi-10. Drivers' due times for this set of instances have been modified by employing different multiplicative factors (1.25, 1.50, 1.75, 2.00, and 3.00). As depicted in Figure 5, by extending due times in a 25%, an average improvement of 11.1% in the shipping time is achieved. This percentage grows up to 15.5% when extending due times in a 50%, and up to 20.8% when extending them in a 300%. In conclusion, defining short shifts in these networks might prove difficult to guarantee solutions with low delivery times. These solutions have been obtained by employing the BR-MS procedure, with a time threshold of 60 seconds, considering the shipping time as the objective to minimize, and repeating three times each execution for each instance and factor (only results for the best-found solutions are reported).

In the next set of experiments, each instance has been executed 10 times, using a different seed for the pseudo-random number generator in each execution. The best results obtained are depicted in Table 5.

Table 4: Summary of instances' characteristics.

Network			Problem		Parameters				Shipments		
Instance	Size	D/H/E	#C	#D	D_{min}	D_{max}	M	L	Long	Short	Mean
pi-01	221	82/83/56	25	210	2	5	2	24	19.947	4.131	9.942
pi-02	382	136/167/79	25	605	3	6	2	24	23.241	2.502	12.94
pi-03	508	192/191/125	25	871	3	6	2	24	23.317	3.169	15.858
pi-04	805	281/316/208	25	1159	2	6	2	24	23.006	2.644	16.173
pi-05	599	200/257/142	25	1383	6	8	2.5	24	23.794	7.304	17.013
pi-06	221	82/89/50	50	1228	10	20	2.5	24	21.159	1.435	12.157
pi-07	382	128/168/86	50	1994	10	20	2	24	23.149	1.98	14.452
pi-08	508	182/194/132	50	2795	10	20	2	24	23.757	2.542	14.182
pi-09	805	272/323/210	50	4756	15	20	3	24	23.77	1.835	14.099
pi-10	599	195/267/137	50	3424	15	20	3	24	23.85	2.51	15.731
pi-11	221	85/88/48	100	1258	10	20	2	12	11.948	2.786	8.124
pi-12	382	131/174/77	100	2279	15	20	3	12	11.997	0.593	7.887
pi-13	508	186/198/124	100	3200	15	20	3	12	11.895	0.296	7.618
pi-14	805	252/322/231	100	3818	10	20	2	12	11.974	1.194	8.136
pi-15	599	214/249/136	100	3736	15	20	3	12	11.967	0.782	8.183
pi-16	221	73/86/62	200	1864	20	30	3	12	11.996	0.548	7.928
pi-17	382	133/167/82	200	3322	20	30	3	12	11.993	0.318	8.047
pi-18	508	174/201/133	200	4333	20	30	3	12	11.962	0.58	7.934
pi-19	805	279/327/199	200	7679	25	30	3	12	11.97	0.216	8.206
pi-20	599	190/279/130	200	4830	20	30	3	12	11.963	0.142	7.901

D: depots. H: hubs. E: endpoints. #C: no. containers; #D: no. drivers.
M: margin for container's due time.
L: maximum length of the shortest path between origin and destination nodes for all containers' shipments (in hours).

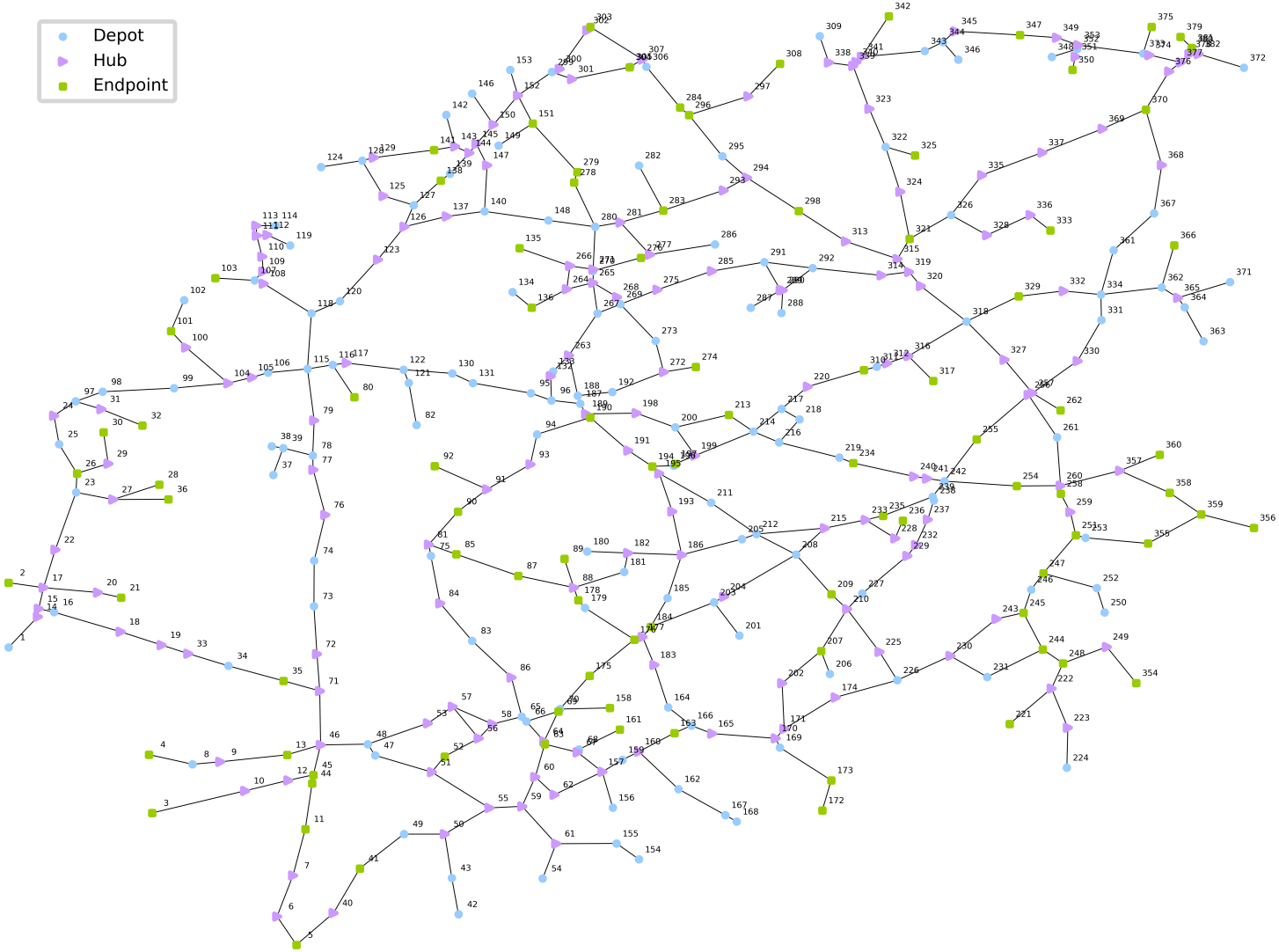

Fig. 4: Representation of the pi-02 instance adapted from TCARP-R2.

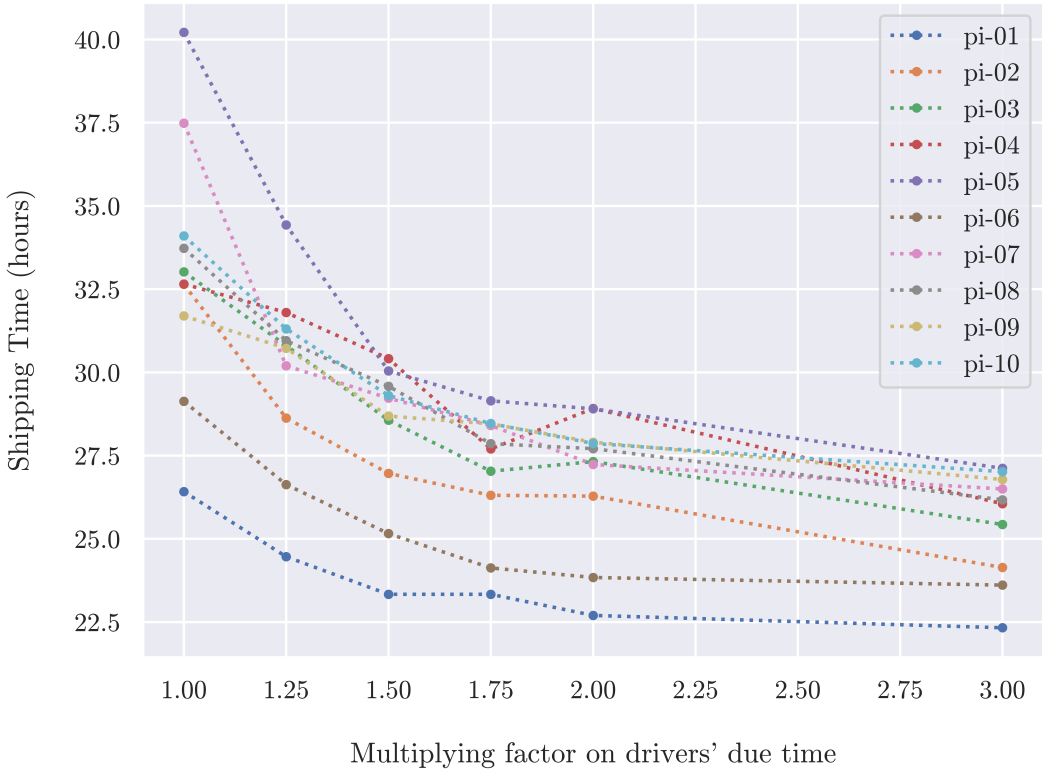

Fig. 5: Shipping time results for instances pi-01 to pi-10 when extending drivers' shifts.

This table also includes: *(i)* the total time (TT) necessary to deliver all containers and return all drivers to their depots; and *(ii)* the shipping time (ST), which does not take into account the time required for the drivers to return home. These last two metrics are reported in hours. Finally,the percentage gaps between all pair combinations of the four approaches are also provided. Table 6 shows the results obtained when the goal is to minimize shipping times (i.e., without including returning times of drivers to their home depots).

Table 5: Performance comparison when optimizing the total time (TT).

	Greedy Heuristic (1)		**BR-MS (2)**		**BR-ILS (3)**		**HILS (4)**		**Gap in TT (%)**					
Instance	TT	ST	TT	ST	TT	ST	TT	ST	(1)-(2)	(1)-(3)	(1)-(4)	(2)-(3)	(2)-(4)	(3)-(4)
pi-01	33.433	29.167	30.175	26.418	30.175	26.418	29.569	27.735	9.74	9.74	11.56	0.00	2.01	2.01
pi-02	36.702	33.672	35.019	32.624	33.455	30.760	34.052	32.267	4.59	8.85	7.22	4.47	2.76	-1.78
pi-03	36.602	33.018	36.602	33.018	36.602	33.018	36.602	33.018	0.00	0.00	0.00	0.00	0.00	0.00
pi-04	40.207	36.329	36.372	33.252	37.151	32.311	35.927	32.893	9.54	7.60	10.64	-2.14	1.22	3.29
pi-05	52.394	48.199	40.737	37.682	42.575	40.331	40.112	37.846	22.25	18.74	23.44	-4.51	1.53	5.79
pi-06	39.137	35.365	32.230	28.458	32.061	28.289	30.721	28.305	17.65	18.08	21.50	0.52	4.68	4.18
pi-07	40.558	38.195	39.856	37.485	39.856	37.485	39.856	37.485	1.73	1.73	1.73	0.00	0.00	0.00
pi-08	39.255	36.345	36.444	33.534	36.637	33.727	35.484	32.480	7.16	6.67	9.61	-0.53	2.63	3.15
pi-09	37.342	35.238	35.100	32.247	35.412	33.308	34.238	32.110	6.00	5.17	8.31	-0.89	2.46	3.32
pi-10	38.326	35.094	37.403	33.494	37.190	34.755	36.816	34.695	2.41	2.96	3.94	0.57	1.57	1.01
pi-11	24.401	21.296	21.528	18.351	21.203	18.210	21.649	18.462	11.77	13.11	11.28	1.51	-0.56	-2.10
pi-12	28.138	25.368	24.939	21.572	25.058	22.288	23.339	20.497	11.37	10.95	17.06	-0.48	6.42	6.86
pi-13	23.382	19.669	22.367	19.825	22.236	18.887	22.451	19.669	4.34	4.90	3.98	0.59	-0.38	-0.97
pi-14	20.694	17.616	20.545	17.616	20.033	17.104	20.394	17.660	0.72	3.19	1.45	2.49	0.73	-1.80
pi-15	25.792	23.146	24.576	21.930	25.052	23.522	23.629	20.597	4.71	2.87	8.39	-1.94	3.85	5.68
pi-16	29.303	24.803	24.741	21.955	25.478	21.740	23.918	21.339	15.57	13.05	18.38	-2.98	3.33	6.12
pi-17	22.966	19.884	22.555	19.622	22.158	19.225	22.487	19.580	1.79	3.52	2.09	1.76	0.30	-1.48
pi-18	26.734	22.038	25.963	22.038	25.909	22.265	25.407	22.060	2.88	3.09	4.96	0.21	2.14	1.94
pi-19	23.123	18.642	22.344	18.642	22.385	18.879	22.767	18.642	3.37	3.19	1.54	-0.18	-1.89	-1.71
pi-20	27.341	24.750	24.428	21.650	25.593	22.114	24.814	22.207	10.65	6.39	9.24	-4.77	-1.58	3.04
Avg.									**7.41**	**7.19**	**8.82**	**-0.32**	**1.56**	**1.83**
Median									**5.36**	**5.78**	**8.35**	**0.00**	**1.55**	**1.98**

TT: total time; ST: shipping time.

Table 6: Performance comparison when optimizing the shipping time (ST).

	Greedy Heuristic (1)		**BR-MS (2)**		**BR-ILS (3)**		**HILS (4)**		**Gap in TT (%)**					
Instance	ST	TT	ST	TT	ST	TT	ST	TT	(1)-(2)	(1)-(3)	(1)-(4)	(2)-(3)	(2)-(4)	(3)-(4)
pi-01	29.167	33.433	26.418	30.175	26.418	30.175	26.418(†)	30.175	9.43	9.43	9.43	0.00	0.00	0.00
pi-02	33.672	36.702	30.875(*)	32.660	32.613	34.403	30.875(†)	34.192	8.31	3.15	8.31	-5.63	0.00	5.33
pi-03	33.018	36.602	33.018	36.602	33.018	36.602	33.018	36.602	0.00	0.00	0.00	0.00	0.00	0.00
pi-04	36.329	40.207	32.837(*)	36.715	32.573	36.451	32.049(*)	35.927	9.61	10.34	11.78	0.80	2.40	1.61
pi-05	48.199	52.394	38.206	41.659	38.533(*)	42.205	39.062	41.306	20.73	20.05	18.96	-0.86	-2.24	-1.37
pi-06	35.365	39.137	28.852	32.495	28.088(*)	31.407	27.824(†)	31.327	18.42	20.58	21.32	2.65	3.56	0.94
pi-07	38.195	40.558	37.485	39.856	36.876(*)	39.247	37.485	39.856	1.86	3.45	1.86	1.62	0.00	-1.65
pi-08	36.345	39.255	33.608	36.518	32.358(*)	35.761	32.480	35.445	7.53	10.97	10.63	3.72	3.36	-0.38
pi-09	35.238	37.342	32.655	37.329	32.314(*)	35.167	32.386	35.546	7.33	8.30	8.09	1.04	0.82	-0.22
pi-10	35.094	38.326	33.237(†)	37.565	33.572(†)	37.573	32.811(†)	37.644	5.29	4.34	6.51	-1.01	1.28	2.27
pi-11	21.296	24.401	18.544	21.693	18.390	21.549	18.351(*)	21.516	12.92	13.65	13.83	0.83	1.04	0.21
pi-12	25.368	28.138	21.249(†)	25.297	22.282(†)	25.229	20.497	23.704	16.24	12.16	19.20	-4.86	3.54	8.01
pi-13	19.669	23.382	18.618(†)	22.754	19.050	22.721	17.881(*)	21.594	5.34	3.15	9.09	-2.32	3.96	6.14
pi-14	17.616	20.694	17.543(†)	20.685	17.252	20.402	17.616(†)	20.469	0.41	2.07	0.00	1.66	-0.42	-2.11
pi-15	23.146	25.792	21.511(*)	24.431	21.930(*)	24.576	21.511	24.109	7.06	5.25	7.06	-1.95	0.00	1.91
pi-16	24.803	29.303	20.990(†)	25.490	21.182(*)	24.792	21.018(†)	25.518	15.37	14.60	15.26	-0.91	-0.13	0.77
pi-17	19.884	22.966	19.575(†)	23.034	19.380	22.693	19.233(†)	22.792	1.55	2.53	3.27	1.00	1.75	0.76
pi-18	22.038	26.734	21.423(*)	24.770	21.649(†)	26.345	21.374(†)	25.913	2.79	1.77	3.01	-1.05	0.23	1.27
pi-19	18.642	23.123	18.281(†)	23.084	18.330(*)	22.075	18.260(†)	23.063	1.94	1.67	2.05	-0.27	0.11	0.38
pi-20	24.750	27.341	20.393(*)	24.512	21.863(*)	25.310	21.379(†)	25.079	17.60	11.66	13.62	-7.21	-4.83	2.21
Avg.									**8.49**	**7.96**	**9.16**	**-0.64**	**0.72**	**1.30**
Median									**7.43**	**6.78**	**8.70**	**-0.14**	**0.17**	**0.77**

Marked with (†) are the shipping times (ST) that are lower than the ones obtained in Table 5 but with greater total times (TT).
Results marked with (*) indicate that both total and shipping times are lower than the ones in the previous table.

7. Analysis of Results

For the 20 instances depicted in Table 5, Figure 6 shows the comparative results for total times. Percentage gaps between each pair of algorithms are used for the comparison. As can be noticed, the BR-MS and BR-ILS approaches have similar performance, with 50% of the gaps between $3 - 11\%$ (1st and 3rd quartile). The hybrid algorithm, HILS, is able to outperform both individual algorithms, offering an average gap around 9% when compared with the DES-based greedy heuristic and about 2% when compared with the BR-MS or the BR-ILS metaheuristics.

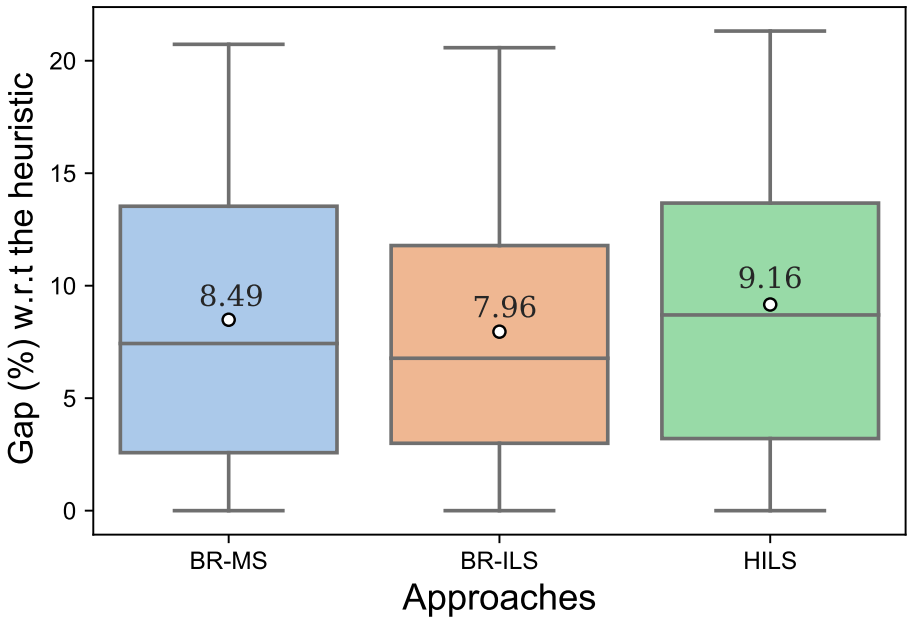

Fig. 6: Performance comparison, in percentage gaps, for total (shipping plus returning) times (TT).

Table 6 shows that the hybrid 2-stage algorithm HILS is the one reporting the best results and the trend is still the same. However, the gaps obtained by the three approaches (with respect to the greedy heuristic) are somewhat greater than the ones from the previous results (Figure 6). This is due to the fact that we are allowing drivers to cover larger distances. The HILS has a slight edge over the rest, with an average gap of 9%, and 1st and 3rd quartiles at 4% and 12%, respectively.

Apart from the total and the shipping time, there are a wide variety of indicators than can be interesting for a manager, e.g.: the number of drivers deployed, the number of hours spent by the drivers, the total time that the drivers were actually driving, etc. The detailed numerical values are given in Table 7, where CT stands for the computational time, #D represents the number of drivers employed, DET corresponds to the drivers total time, and DTT refers to the drivers traveling time. All units are in hours, except for the number of drivers and the computational time (this one is given in seconds). Notice that the data of the radar plots has been reversed, i.e.: optimizing a feature means having the greatest value, 100%.

(a) Greedy heuristic solution characteristics.

(b) BR-MS solution characteristics.

(c) BR-ILS solution characteristics.

(d) HILS solution characteristics.

Fig. 7: A graphical representation of different solutions on multiple dimensions (indicators).

For instance, the solution provided by the deterministic heuristic is the one employing the minimum computational time. In this regard, as it is shown in Figure 7, the solution obtained by the HILS approach is the one yielding the minimum total time, although the minimum shipping time was obtained by the BR-ILS algorithm by using 4 more drivers. However, the solution found by means of the BR-MS is the most balanced one.

Table 7: Measuring different indicators associated with four solutions (instance *pi-06*).

Approach	TT (h)	ST (h)	CT (s)	#D	DET (h)	DTT (h)
Greedy Heuristic	39.14	35.37	0	189	1677	1349
BR-MS	32.23	28.46	66	190	1671	1316
BR-ILS	31.41	28.09	42	198	1722	1349
HILS	30.72	28.31	147	194	1714	1335

8. Conclusions and Future Work

Real-life transport management in global networks can drastically scale up in difficulty with increasing customers' demands and shipping volumes. In those situations, managers require from intelligent algorithms to properly address the underlying large-scale routing optimization problems. In the context of interconnected networks with intermediate hubs that can temporarily storage containers, this paper introduces a 'discrete-event' heuristic as a first approach to solve a coordinated multi-vehicle container transport problem. By combining concepts from discrete-event simulation with a greedy heuristic, it is possible to effectively deal with the challenging synchronization issues that arise in these dynamic systems, which evolve over time as new decisions are made in response to the emerging events. Thus, in a network of containers, depots, drivers, hubs, and endpoints, our main goal is to find routing solutions that minimize the total time required to: *(i)* deliver the containers to their final destinations before their deadline expires; and *(ii)* allow the drivers to return to their home depots before their working shift is over. Apart from this main goal, other indicators (such as the number of employed drivers, the total driving time, etc.) are also reported in a multi-dimensional graph. For small-case instances, the proposed heuristic approach yields optimal solutions within much less execution time than the one required by the exact model.

The former discrete-event-based heuristic is extended into a probabilistic algorithm by introducing biased-randomization techniques. Hence, a biased-randomized multi-start procedure is proposed as a way to generate multiple solutions with different characteristics. The randomization of the heuristic is performed every time a decision needs to be made regarding: *(i)* the next container to be dispatched; *(ii)* the next driver to be activated; and *(iii)* the next node that the container will visit in the path to its final destination. For the tested instances, this algorithm is able to outperform the initial greedy heuristic by a gap of 7% on the average. Next, in order to intensify the local search around 'good' base solutions, the biased-randomization version of the heuristic is also extended into a iterated local search procedure. According to the computational experiments, the performance of this second metaheuristic is quite similar to that of the multi-start one. Finally, both the multi-start (exploration-oriented) and the iterated local search (intensification-oriented) procedures are combined into a 2-stage algorithm. Using the same computational times, this hybrid algorithm is able to slightly outperform the previous metaheuristics, with an average gap of 9% with respect to the initial greedy heuristic.

This paper introduces the concept of discrete-event heuristic: a solution-construction procedure that incorporates efficient decision-making criteria inside a deterministic discrete-event simulation. By combining such a discrete-event greedy heuristic with the concept of biased randomization, our approach

is able to quickly generate feasible and high-quality solutions to a challenging transportation problem. This allows us to deal with large-size instances that cannot be efficiently solved by just employing exact methods or other metaheuristic algorithms unless they take into account the cause-and-effect relations among the chronological events. By introducing this novel approach, new transportation strategies might be considered by supply chain managers. Hence, by increasing the level of coordination and synchronization among drivers, managers can propose scenarios in which these drivers do not have to cover large distances, while still minimizing the total time required to complete the transportation of containers. Probably the main limitation of the proposed approach is the difficulty to measure the quality of its solutions in large-size instances, where exact methods cannot provide optimal values and where other metaheuristics might also have severe limitations due to the feasibility constraints imposed by the time-dependent events that characterize this highly dynamic transportation environment.

In our view, this work has several research lines that are worthy to be explored in the future, among them: *(i)* to extend the hybrid algorithm into a simulation-optimization one (Chica et al., 2020), so it can deal with real-life uncertainty (e.g., stochastic travel times, stochastic loading / unloading times, etc.); *(ii)* to analyze the effect of relaxing the deadlines (both for containers and drivers) in the quality of the solutions, specially considering that future vehicles might not require a human driver; *(iii)* to combine this algorithm with machine learning methods (Arnau et al., 2018; Bayliss et al., 2020), so that it can be used to solve optimization problems with dynamic inputs (e.g. the travel times or storage capacities might depend upon the previous routing decisions); and *(iv)* to incorporate economic and environmental costs in the objective function as well as to analyze tradeoffs between total distribution time and economic cost –e.g., by considering a penalty cost in case of delays.

Acknowledgments

This work has been partially supported by the Spanish Ministry of Science (PID2019-111100RB-C21 / AEI / 10.13039/501100011033, RED2018-102642-T), and the Erasmus+ program (2019-I-ES01-KA103-062602).

References

Arnau, Q., Juan, A.A., Serra, I., 2018. On the use of learnheuristics in vehicle routing optimization problems with dynamic inputs. *Algorithms* 11, 12, 208.

Bayliss, C., Juan, A.A., Currie, C.S., Panadero, J., 2020. A learnheuristic approach for the team orienteering problem with aerial drone motion constraints. *Applied Soft Computing* 0, 0, 106280.

Belloso, J., Juan, A.A., Faulin, J., 2019. An iterative biased-randomized heuristic for the fleet size and mix vehicle-routing problem with backhauls. *International Transactions in Operational Research* 26, 1, 289–301.

Bianchessi, N., Mansini, R., Speranza, M.G., 2018. A branch-and-cut algorithm for the team orienteering problem. *International Transactions in Operational Research* 25, 2, 627–635.

Chambers, M., Goworowska, J., Rick, C., Sedor, J., 2015. *Freight facts and figures 2015*. U.S. Department of Transportation. Bureau of Transportation Statistics.

Chen, X., Wang, X., 2016. Effects of carbon emission reduction policies on transportation mode selections with stochastic demand. *Transportation Research Part E: Logistics and Transportation Review* 90, 196–205.

Chica, M., Juan, A.A., Bayliss, C., Cordón, O., Kelton, W.D., 2020. Why simheuristics? benefits, limitations, and best practices

when combining metaheuristics with simulation. *SORT-Statistics and Operations Research Transactions* xx, 311–334.
Costello, B., Suarez, R., 2015. Truck driver shortage analysis 2015. *American Trucking Associations* 206, 1–12.
De Armas, J., Keenan, P., Juan, A.A., McGarraghy, S., 2019. Solving large-scale time capacitated arc routing problems: from real-time heuristics to metaheuristics. *Annals of Operations Research* 273, 1-2, 135–162.
Díaz, M., Martín, C., Rubio, B., 2016. State-of-the-art, challenges, and open issues in the integration of internet of things and cloud computing. *Journal of Network and Computer Applications* 67, 99–117.
Dominguez, O., Juan, A.A., De La Nuez, I., Ouelhadj, D., 2016. An ils-biased randomization algorithm for the two-dimensional loading hfvrp with sequential loading and items rotation. *Journal of the Operational Research Society* 67, 1, 37–53.
Dominguez, O., Juan, A.A., Faulin, J., 2014. A biased-randomized algorithm for the two-dimensional vehicle routing problem with and without item rotations. *International Transactions in Operational Research* 21, 3, 375–398.
El-Hajj, R., Dang, D.C., Moukrim, A., 2016. Solving the team orienteering problem with cutting planes. *Computers and Operations Research* 74, C, 21–30.
Fazili, M., Venkatadri, U., Cyrus, P., Tajbakhsh, M., 2017. Physical internet, conventional and hybrid logistic systems: A routing optimisation-based comparison using the eastern canada road network case study. *International Journal of Production Research* 55, 9, 2703–2730.
Ferone, D., Gruler, A., Festa, P., Juan, A.A., 2019. Enhancing and extending the classical GRASP framework with biased randomisation and simulation. *Journal of the Operational Research Society* 70, 8, 1362–1375.
Ferone, D., Hatami, S., González-Neira, E.M., Juan, A.A., Festa, P., 2020. A biased-randomized iterated local search for the distributed assembly permutation flow-shop problem. *International Transactions in Operational Research* 27, 3, 1368–1391.
Ferrer, A., Guimarans, D., Ramalhinho, H., Juan, A.A., 2016. A BRILS metaheuristic for non-smooth flow-shop problems with failure-risk costs. *Expert Systems with Applications* 44, 177–186.
Fikar, C., Juan, A.A., Martinez, E., Hirsch, P., 2016. A discrete-event driven metaheuristic for dynamic home service routing with synchronised trip sharing. *European Journal of Industrial Engineering* 10, 3, 323–340.
Floyd, R.W., 1962. Algorithm 97: Shortest path. *Communications of the ACM* 5, 6, 345.
Funke, J., Kopfer, H., 2016. A model for a multi-size inland container transportation problem. *Transportation Research Part E: Logistics and Transportation Review* 89, 70–85.
Gendron, B., Crainic, T.G., Frangioni, A., 1999. Multicommodity capacitated network design. In *Telecommunications network planning*. Springer, pp. 1–19.
Goel, A., 2018. Legal aspects in road transport optimization in europe. *Transportation research part E: logistics and transportation review* 114, 144–162.
Goel, A., Vidal, T., Kok, A.L., 2020. To team up or not: single versus team driving in european road freight transport. *Flexible Services and Manufacturing Journal* xx, x, 1–35.
Grasas, A., Juan, A.A., Faulin, J., de Armas, J., Ramalhinho, H., 2017. Biased randomization of heuristics using skewed probability distributions: a survey and some applications. *Computers & Industrial Engineering* 110, 216–228.
Gubbi, J., Buyya, R., Marusic, S., Palaniswami, M., 2013. Internet of things (IoT): A vision, architectural elements, and future directions. *Future generation computer systems* 29, 7, 1645–1660.
Hakimi, D., Montreuil, B., Hajji, A., 2015. Simulating physical internet enabled hyperconnected semi-trailer transportation systems. In *Proceedings of 2nd International Physical Internet Conference, Paris, France*.
Hao, C., Yue, Y., 2016. Optimization on combination of transport routes and modes on dynamic programming for a container multimodal transport system. *Procedia Engineering* 137, 382–390.
He, J., Huang, Y., Chang, D., 2015. Simulation-based heuristic method for container supply chain network optimization. *Advanced Engineering Informatics* 29, 3, 339–354.
Hsu, C.I., Hsieh, Y.P., 2007. Routing, ship size, and sailing frequency decision-making for a maritime hub-and-spoke container network. *Mathematical and Computer Modelling* 45, 7, 899–916.
Ji, M., Shen, L., Shi, B., Xue, Y., Wang, F., 2015. Routing optimization for multi-type containerships in a hub-and-spoke network. *Journal of Traffic and Transportation Engineering* 2, 5, 362–372.
Ji, M.J., Chu, Y.L., 2012. Optimization for hub-and-spoke port logistics network of dynamic hinterland. *Physics Procedia* 33, 827–832.
Juan, A.A., Faulin, J., Ferrer, A., Lourenço, H.R., Barrios, B., 2013. Mirha: multi-start biased randomization of heuristics with adaptive local search for solving non-smooth routing problems. *Top* 21, 1, 109–132.

Juan, A.A., Lourenço, H.R., Mateo, M., Luo, R., Castella, Q., 2014. Using iterated local search for solving the flow-shop problem: parallelization, parametrization, and randomization issues. *International Transactions in Operational Research* 21, 1, 103–126.

Keenan, P., 2017. TCARP large rural datasets. https://www.researchgate.net/publication/320386608_tcarp_large_rural_datasets.

Kopetz, H., 2011. Internet of things. In *Real-time systems*. Springer, pp. 307–323.

Lourenço, H.R., Martin, O.C., Stützle, T., 2010. Iterated Local Search: Framework and Applications, Springer US, Boston, MA. pp. 363–397.

Magnanti, T.L., Wong, R.T., 1984. Network design and transportation planning: Models and algorithms. *Transportation Science* 18, 1, 1–55.

Martí, R., Resende, M.G., Ribeiro, C.C., 2013. Multi-start methods for combinatorial optimization. *European Journal of Operational Research* 226, 1, 1–8.

Meller, R.D., Ellis, K., Loftis, B., 2012. From horizontal collaboration to the physical internet: Quantifying the effects on sustainability and profits when shifting to interconnected logistics systems. *Center for Excellence in Logistics and Distribution, University of Arkansas*

Montreuil, B., 2011. Toward a physical internet: Meeting the global logistics sustainability grand challenge. *Logistics Research* 3, 2-3, 71–87.

Neves-Moreira, F., Amorim, P., Guimarães, L., Almada-Lobo, B., 2016. A long-haul freight transportation problem: Synchronizing resources to deliver requests passing through multiple transshipment locations. *European Journal of Operational Research* 248, 2, 487–506.

Pagès-Bernaus, A., Ramalhinho, H., Juan, A.A., Calvet, L., 2019. Designing e-commerce supply chains: a stochastic facility–location approach. *International Transactions in Operational Research* 26, 2, 507–528.

Pan, S., Ballot, E., Huang, G.Q., Montreuil, B., 2017. Physical internet and interconnected logistics services: research and applications. *International Journal of Production Research* 55, 9, 2603–2609.

Quintero-Araujo, C.L., Caballero-Villalobos, J.P., Juan, A.A., Montoya-Torres, J.R., 2017. A biased-randomized metaheuristic for the capacitated location routing problem. *International Transactions in Operational Research* 24, 5, 1079–1098.

Sallez, Y., Pan, S., Montreuil, B., Berger, T., Ballot, E., 2016. On the activeness of intelligent physical internet containers. *Computers in Industry* 81, 96–104.

Sarraj, R., Ballot, E., Pan, S., Hakimi, D., Montreuil, B., 2014. Interconnected logistic networks and protocols: Simulation-based efficiency assessment. *International Journal of Production Research* 52, 11, 3185–3208.

Serrano-Hernández, A., Juan, A.A., Faulin, J., Perez-Bernabeu, E., 2017. Horizontal collaboration in freight transport: concepts, benefits and environmental challenges. *SORT-Statistics and Operations Research Transactions* 1, 2, 393–414.

Sohrabi, H., Montreuil, B., Klibi, W., 2016. Collaborative and hyperconnected distribution systems: A comparative optimization-based assessment. In *Proceedings of Proceedings of the 2016 Industrial and Systems Engineering Research Conference, Anaheim, CA*.

Treiblmaier, H., Mirkovski, K., Lowry, P.B., Zacharia, Z.G., 2020. The physical internet as a new supply chain paradigm: a systematic literature review and a comprehensive framework. *The International Journal of Logistics Management*

Wang, C.x., 2008. Optimization of hub-and-spoke two-stage logistics network in regional port cluster. *Systems Engineering-Theory & Practice* 28, 9, 152–158.

White, K.P., Ingalls, R.G., 2015. Introduction to simulation. In *Proceedings of the 2015 Winter Simulation Conference*, IEEE, pp. 1741–1755.

Wolfinger, D., Salazar-González, J.J., 2021. The pickup and delivery problem with split loads and transshipments: A branch-and-cut solution approach. *European Journal of Operational Research* 289, 2, 470–484.

Wolfinger, D., Tricoire, F., Doerner, K.F., 2019. A matheuristic for a multimodal long haul routing problem. *EURO Journal on Transportation and Logistics* 8, 4, 397–433.

Zäpfel, G., Wasner, M., 2002. Planning and optimization of hub-and-spoke transportation networks of cooperative third-party logistics providers. *International journal of production economics* 78, 2, 207–220.