A CP Approach for the Liner Shipping Network Design Problem

Yousra El Ghazi
Aix Marseille Univ, Université de Toulon, CNRS, LIS, Marseille, France

Djamal Habet
Aix Marseille Univ, Université de Toulon, CNRS, LIS, Marseille, France

Cyril Terrioux
Aix Marseille Univ, Université de Toulon, CNRS, LIS, Marseille, France

Abstract

The liner shipping network design problem consists, for a shipowner, in determining, on the one hand, which maritime lines (in the form of rotations serving a set of ports) to open, and, on the other hand, the assignment of ships (container ships) with the adapted sizes for the different lines to carry all the container flows. In this paper, we propose a modeling of this problem using constraint programming. Then, we present a preliminary study of its solving using a state-of-the-art solver, namely the OR-Tools CP-SAT solver.

2012 ACM Subject Classification Computing methodologies → Artificial intelligence

Keywords and phrases Constraint optimization problem, modeling, solving, industrial application

Digital Object Identifier 10.4230/LIPIcs.CP.2023.16

Supplementary Material *Dataset*: `https://pageperso.lis-lab.fr/cyril.terrioux/LSNDP/instances.zip`

Funding This work has been funded by Bpifrance under the PIA project Transformation Numérique du Transport Maritime (TNTM).

Acknowledgements The authors would like to thank the anonymous reviewers for their valuable comments and suggestions.

1 Introduction

Nowadays, maritime transport plays a major role in world trade. According to the International Maritime Organization (IMO), more than 80% of international trade is carried out by sea. The transport of containerized commodities constitutes the major part of this trade. It relies on more than 5,000 container ships that serve more than 500 ports worldwide. In this context, many combinatorial optimization problems [6, 9, 28] may arise with non-negligible economic and ecological impacts given their scale.

In this paper, we focus on the Liner Shipping Network Design Problem (LSNDP [9]). A shipping line, also called a service, is defined by a cyclic route (called a rotation) that visits a given set of ports in a given order and at regular times (see, for example, Figure 1). Generally, each port is thus visited by a vessel of the line at a weekly or biweekly frequency. All the vessels on a line are assumed to be homogeneous in terms of their main features (loading capacity, speed, fuel consumption, engines, ...). Operating a weekly line with a rotation lasting k weeks requires k vessels of the same type. Given a set of ports, a fleet of container ships, and a container flow (defined by a set of triples consisting of the original port of the commodities, their destination port, and the number of containers they represent), the LSNDP problem consists, for a shipowner, in determining, on the one hand, which shipping lines to open, and, on the other hand, which ships to operate on each line in order to carry as

29th International Conference on Principles and Practice of Constraint Programming (CP 2023).
Editor: Roland H. C. Yap; Article No. 16; pp. 16:1–16:21
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

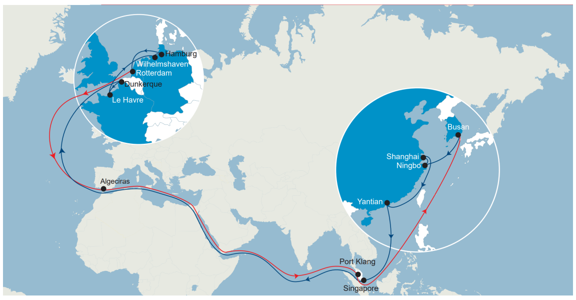


Figure 1 Example of a line connecting Asia and Europe.

many commodities as possible while ensuring a weekly frequency of visits to each port and optimizing costs. It is classified as NP-hard [7]. To give an idea of its difficulty, we can note that, taken separately, each of its two subproblems is already an NP-hard problem [7, 13]. Moreover, from a practical viewpoint, its solving by exact methods is currently limited to instances with a dozen ports at most. Although recent, this problem has been the subject of many works, especially in the last ten years. Most of them are from operational research. Note that, in the literature, different variants of the LSNDP problem are studied according to the assumptions and properties taken into account (transit time, transshipment, constant or variable speed from one rotation to another or from one leg (i.e. a trip between two consecutive ports in the rotation) to another, type of service, possibility of refusing some commodities, ...).

Different approaches have been considered, often based on integer (mixed) linear programming (e.g. [18, 19, 21, 22, 23, 27, 30]). They are mainly based on two types of formulations. The first type of formulation is service-oriented. The set of possible services is calculated upstream and provided as input to the model. The latter is then limited to selecting the services to be kept among the candidates. The main disadvantage of this type of formulation is that the number of possible services grows exponentially with the number of ports, which limits its practical interest in the context of a solving performed with complete methods. On the other hand, it can be interesting in the context of incomplete methods, because one can then consider only a subset of the possible services. In practice, the solution proposed in [2] and based on a tabu method coupled with column generation allowed handling instances up to 120 ports. The second type of formulation is based on the selection of the arcs of the graph representing the possible links between each pair of ports. A service is then defined by the arcs that compose it, and the same arc can be used to define several services. From a practical viewpoint, such modeling coupled with complete methods [18, 23, 22] allows handling instances with up to a dozen ports [9].

Other approaches (e.g. [1, 2]) are based on two-step solving. Since the LSNDP problem consists of two subproblems, they process each subproblem separately. For example, the approaches presented in [2, 5, 13, 12] solve, in the first step, the problem of creating services and, in the second step, consider the vessel assignment and the management of the commodity flow based on the services found by the first step. In [17], the first phase is devoted to the management of the flow, while the second defines the services. Generally, the solving is done in several passes, the first phase then benefiting from some feedbacks from the second phase of the previous pass. Of course, this type of approach corresponds to incomplete methods. In practice, these approaches provide satisfactory results for instances with up to 120 ports [2]. They are often based on matheuristics (e.g. [5, 13, 12]) or the Variable Neighborhood Search algorithm (VNS) such as [17].

Beyond that, there are many related problems to the LSNDP problem. For example, the Vehicle Routing Problem (VRP [15]) and its variants have strong similarities with the LSNDP problem. In particular, the routes are circuits and, for some variants, the vehicle load or transit times can be taken into account. In maritime transport, the Liner Shipping Fleet Repositioning Problem (LSFRP [28]) consists in moving container ships from one service to another while taking into account the commodities to be transported, the empty containers to be relocated and maximizing the difference between the revenues and the costs generated. Among the approaches studied to solve this problem, we can underline the interest in using constraint programming (CP) put forward in [14].

While the VRP and LSFRP problems (and other shipping-related problems such as [16, 24]) have been studied from a CP perspective, this does not seem to be the case for the LSNDP problem. In this paper, we propose a model to handle a relatively general version of the LSNDP problem. Our model considers variable speeds from one trip to another and takes into account transshipments and transit times. Although developed in partnership with one of the world's leading container shipping companies, the model presented takes into account a relatively general version of the LSDNP problem. It can, of course, be adapted to specific needs, taking advantage of the flexibility of constraint programming. One of the aims of this work is to study the interest of a CP-based approach to modeling and solving such problems.

This paper is organized as follows. Section 2 introduces the notions necessary to understand the paper. Then, in Section 3, we propose a CP model for the LSNDP problem. Finally, we present some experimental results, in Section 4, before discussing related work in Section 5 and concluding in Section 6.

2 Preliminary

2.1 Constraint Programming

An instance P of the Constraint Optimization Problem (COP) can be defined as a 4-tuple (X, D, C, f) where $X = \{x_1, \dots, x_n\}$ is the set of *variables*, $D = \{D_{x_1}, \dots, D_{x_n}\}$ is the set of domains, the domain D_{x_i} being related to the variable x_i, $C = \{c_1, ..., c_e\}$ represents the set of the *constraints* which define the interactions between the variables and describe the allowed combinations of values and f specifies the criterion to be optimized. Solving a COP instance $P = (X, D, C, f)$ amounts to finding an assignment of all variables of X satisfying all constraints of C and optimizing the criterion given by f. This is an NP-hard problem.

One of the advantages of constraint programming lies in the existence of specialized constraints (the *global constraints*) which will make easier the modeling of problems, but also, their solving thanks to their dedicated filtering algorithms. In the following, we will exploit the following global constraints (where $\odot$ denotes a relational operator among $\leq$, $<$, $=$, $\neq$, $>$ or $\geq$):

- `Alldiff-except`(Y, v) [3, 10] which ensures that the values of the variables of Y are pairwise distinct, except in the case where they are equal to v,
- `Circuit`(Y, ℓ) [4] which imposes that the values of the variables of Y form a circuit of length ℓ (in the sense of graph theory), each variable y_i having for value i if it does not take part in the circuit, and j (with $i \neq j$) if j is the successor of i in the circuit,
- `Count`$(Y, V) \odot k$ [4, 8] which ensures that the number of variables of Y whose value belongs to V satisfies the condition imposed by the relation $\odot$ with respect to k,
- `Elt`$(Y, i) = k$ [29] which ensures that the ith value of Y (using a 0-based indexing) is equal to k (Y can be here an ordered set of variables or values),

- $\texttt{Elt}_\texttt{m}(Y, i, j) = k$ [4, 3] which ensures the same property as `Elt`, but, for an orderede set Y of variables or values organized in the form of a two-dimensional matrix,
- $\texttt{Maximum}(Y) = k$ which ensures that the greatest value of Y is equal to k (Y can here be a set of variables or expressions),
- $\texttt{Sum}(Y, \Lambda) \odot k$ which imposes that the sum of the values of Y weighted by the coefficients of Λ satisfies the condition imposed by the relation $\odot$ with respect to k. In the following, this constraint will be represented in the more explicit form $\sum_i \lambda_i.y_i \odot k$.

2.2 The Liner Shipping Network Design Problem

Liner shipping involves the use of standardized vessels that will reliably move cargo between ports according to a pre-determined route and schedule. It is often compared to scheduled passenger service, such as a train or bus service because it operates on a fixed schedule and provides regular and predictable service for shippers and receivers of commodities. A shipping line, also called a service, is defined by a cyclic route (called a rotation) that serves a set of ports in a specific order and on a regular schedule. Figure 1 describes the example of a line connecting Asia and Europe.

In this paper, we consider only the transportation of commodities in their containerized form, as this mode of transportation constitutes the bulk of freight transportation in terms of quantity and value. Thus, a customer wishing to move commodities from a port of loading (POL) to a port of destination (POD) needs to place them inside one or more containers. Containers have the advantage, for the shipowner, that their dimensions are standardized, thus facilitating their handling and placement on board of specialized vessels such as container ships. There are mainly two sizes of containers: 20-foot containers (about 6.1 m) and 40-foot containers (12.2 m) with a height of 8.6 feet (2.6 m) and a width of 8 feet (2.4 m). The majority of containers transported are of one of these two sizes. Also, the storage space of the vessels is divided into 40-foot unit spaces on which it is possible to stack both 40-foot and 20-foot units. The Twenty-foot Equivalent Unit (TEU) is the unit generally used to count a number of containers. For example, a 40-foot container counts as 2 TEUs.

From the carrier's viewpoint, each commodity k is seen as a quantity $q(k)$ of containers (expressed in TEUs) to be transported from the port of origin $pol(k)$ to the port of destination $pod(k)$ in exchange for a revenue $rev(k)$ per TEU (expressed in dollars). This revenue may be zero in the case of empty containers. Some commodities may have a maximal transit time $tt_{max}(k)$ that must be respected. This time is the maximum time allowed for their transport. Generally, such commodities are transported within the framework of premium offers proposed by the carriers. It should be noted that a batch of containers sent by a customer from one port to another cannot be divided into several sub-batches. Finally, cargo can be transported from its port of origin to its port of destination via the successive use of different lines. The operation of unloading a commodity from one line and loading it on another is called "*transshipment*". It may require the commodities to be stored for several days in the transshipment port until the vessel of the next line arrives and loads them on board. This can result in costs (see transshipment costs below) and longer travel times.

Concerning vessels, container ships are grouped by type of vessel with identical or similar features. Thus, each class v is characterized by its *capacity* $\kappa(v)$ (i.e. the maximum number of containers (in TEU) that can be transported), its *daily charter rate* $tc(v)$ (corresponding to the daily cost of using the vessel), its *interval of possible speeds* $[\nu_{min}(v), \nu_{max}(v)]$ (in knots), its *hourly consumption* $cons(v, \nu)$ *of fuel* for the main engine (in tons per hour), for each *type of fuel* $fuel(v)$ and each possible speed ν. Regarding consumption, other parameters that could have an impact such as wind strength, sea currents, draft or load on board are ignored. These parameters can be variable in time and difficult to anticipate, as the lines are generally defined on a yearly scale.

Each port p also has its own features, namely its *productivity* $prod(p, v)$ (i.e., the number of containers loaded or unloaded per hour for vessels of type v), its waiting time $wt(p, v)$ (time at anchor before entering the port), its maneuvring time $man^{in}(p, v)$ and $man^{out}(p, v)$ to enter and leave the port respectively, its call charges $pc(p, v)$ (in dollars), and its transshipment cost $ts(p)$ (in dollars). The times are given in hours and depend on the type v of vessels, as do the call charges. A canal c (e.g. the Suez or Panama Canal) is characterized by a waiting time $wt(c, v)$, a traversal time $trav(c)$ (in hours), and a traversal cost $pc(c, v)$ (in dollars).

The number of vessels operating on a service is determined by the length of the rotation and the frequency of departures. Indeed, a rotation must guarantee a regular frequency of visits to each port it serves. This rotation frequency is generally weekly or biweekly. For a weekly frequency, the duration of the rotation must be a multiple of seven days. The number of vessels deployed per rotation must then be equal to the number of weeks in that rotation. For example, the line shown in Figure 1 has a duration of 91 days or 13 weeks. It is therefore operated with 13 vessels.

The liner shipping network design problem (LSNDP) can be defined as follows: Given a set of *ports*, a set of *vessels* divided by type (each type v having $nb(v)$ vessels) and a set of *commodities* to be transported, define a set of rotations having a weekly frequency and determine the vessels operating them to transport the commodities while respecting, if necessary, the transit times and maximizing the profit. The profit is defined as the difference between the revenues generated by the commodities transported and all the costs generated by this transport (fuel costs, vessel operating costs, port call and canal costs, transshipment costs, . . .). To calculate fuel costs, for each type of fuel f, we have the price $fp(f)$ per ton of fuel (expressed in dollars).

While the primary purpose of this problem is to design maritime transportation networks, it can also be used to assist in decision-making. For example, it can be used to simulate situations such as traffic jams to enter certain ports and determine whether or not it is relevant to adapt existing rotations. It can also be used to consider changes in the flow of containers to be transported, to evaluate the interest in taking market share in certain commodity flows or to anticipate the construction of new ships.

3 Model

3.1 Modeling Choices

In our model, we adopt the usual assumptions of the literature. In particular, we assume that all container ships of a given type have identical features and that the frequency of services is weekly. Furthermore, we choose to treat canals (such as the Suez and Panama canals) in the same way as ports. The time it takes to cross a canal replaces the time it takes to load/unload a ship in a port. As a result, the notion of rotation now also takes canals into account. Since a rotation can pass several times through the same canal, but not through the same port, we consider, in our model, two instances of each canal so that a canal can be used both on the "outbound" and on the "return". For example, we can see that the line represented in Figure 1 passes twice through the Suez Canal, once on the "outbound" (blue route) and once on the "return" (red route). Note that creating more than two instances of the same canal is of little interest because a solution passing more than twice through the same canal has little chance of being optimal.

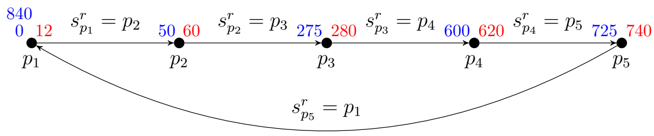


Figure 2 A rotation involving five ports.

$$\texttt{Circuit}(\{s^r_p \mid p \in \mathcal{P} \cup \mathcal{C}\}, \ell_r) \qquad r \in \mathcal{R} \qquad \text{(R.1)}$$

$$v_r > 0 \Rightarrow \sum_{p \in \mathcal{P}} (s^r_p \neq p) \geq 3 \qquad r \in \mathcal{R} \qquad \text{(R.2)}$$

$$v_r > 0 \iff \ell_r \geq 3 \qquad r \in \mathcal{R} \qquad \text{(R.3)}$$

Figure 3 Constraints related to rotations and routes.

Our model takes, as inputs, all the information about the ports and canals, the flow of commodities, the types of vessels, and the distances between pairs of ports/canals. It also relies on the maximum number r_{max} of rotations to be defined, the time horizon h_{max} (in hours), that is the maximum duration to achieve a rotation, and the maximum number ts_{max} of transshipments allowed per commodity. One of the particularities of this model is that the main operations will be time-stamped to handle rotation or transit times of the commodities as precisely as possible. Moreover, speeds can be different from one leg to another. Finally, it takes into account the possibility of refusing to transport a commodity in the network if it is not profitable or impossible.

Thereafter, given the large number of variables, we define the variables progressively when needed. The set of ports is denoted $\mathcal{P} = \{0, 1, \ldots, |\mathcal{P}| - 1\}$, the set of canals $\mathcal{C} = \{c, c + |\mathcal{C}_0| \; s.t. \; c \in \mathcal{C}_0\}$ (with $\mathcal{C}_0 = \{|\mathcal{P}|, \ldots, |\mathcal{P}| + |\mathcal{C}_0| - 1\}$ the set of canals before duplication), the set of type of vessels available $\mathcal{V} = \{1, 2, \ldots, |\mathcal{V}|\}$ and that of the commodities $\mathcal{K} = \{1, 2, \ldots, |\mathcal{K}|\}$. We note respectively $\mathcal{I}$ and $\mathcal{I}^+$ the index sets $[0, ts_{max}]$ and $[0, ts_{max} + 1]$. Let $\mathcal{R} = \{1, \ldots, r_{max}\}$ be the set of the indices of the possible rotations.

3.2 Definition of Rotations

Our model does not necessarily use all possible r_{max} rotations. Therefore, we consider a variable v_r per rotation r. Its value is an integer between 1 and $|\mathcal{V}|$ representing the type of vessels exploited if the rotation is used, 0 otherwise. Each rotation r must correspond to a circuit. To define such circuits, we introduce a variable s^r_p per port/canal p and rotation r. Its value is p if the port/canal p is not involved in the circuit, the successor of port/canal p otherwise. Figure 2 illustrates this for a rotation involving five ports. We also introduce a variable ℓ_r specifying the length of the circuit associated with rotation r. Thus, for each rotation r, the existence of a circuit can be guaranteed by constraint (R.1) (see Figure 3). Note that this constraint avoids the existence of subtours, a property that is not generally easy to guarantee. For instance, in MIP, avoiding subtours requires adding non-linear constraints that must then be linearized. Then, thanks to constraint (R.2), a circuit must involve at least three ports (this is a business rule generally desired by carriers), and so, cannot involve only canals. Finally, constraint (R.3) ensures that rotation r is used if and only if the associated circuit has a length at least equal to three.

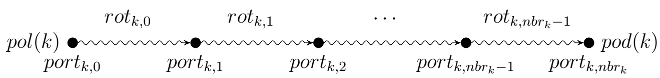


Figure 4 Transport of a commodity k from its origin port $pol(k)$ to its destination port $pod(k)$.

3.3 Cargo Flow

Our model allows for the possibility of not carrying a commodity k if it is not possible or not profitable. To do this, we define a Boolean variable α_k that is true if commodity k is accepted in the network, false otherwise. Taking charge of commodity k means that it is loaded at the original port $pol(k)$ and unloaded at the destination one $pod(k)$, possibly passing through intermediate ports. In our model, we consider only the intermediate ports where the commodity will be transshipped as shown in Figure 4. Also, we introduce a variable $rot_{k,i}$ per commodity k and step i to represent the ith rotation used to transport the commodity k. $rot_{k,i}$ has the value r if commodity k is carried thanks to rotation r during the ith step, -1 if this step is not needed. Similarly, the variable $port_{k,i}$ represents the port where the commodity k enters its ith rotation. By so doing, commodity k enters in its ith rotation at port $port_{k,i}$ and leaves it at port $port_{k,i+1}$ (i.e. the port in which it starts its next rotation if $port_{k,i+1}$ differs from $pod(k)$). These variables have the value of the corresponding port p ($p \in \mathcal{P}$) if the ith rotation is used, -1 otherwise. More precisely, the domain of $port_{k,i}$ is $\{-1, pol(k)\}$ if $i = 0$, $\{-1, pod(k)\}$ if $i = ts_{max} + 1$, $\{-1\} \cup \mathcal{P} - \{pol(k)\}$ otherwise. For each commodity k, a variable nbr_k specifies the number of rotations used (between 0 and $ts_{max} + 1$).

We can now define the associated constraints (see Figure 5). First, constraint (F.1) specifies that commodity k is accepted in the network if and only if there is at least one rotation that carries it. Of course, the port of departure of an accepted commodity k is necessarily its port of origin $pol(k)$ (constraint (F.2)). The last port used is necessarily the destination port $pod(k)$. To ensure this, we introduce a variable pod_k per commodity k that can take two values: either $pod(k)$ if commodity k is accepted, or -1 otherwise. Constraint (F.3) guarantees that pod_k has the relevant value depending on the value of α_k while constraint (F.4) ensures that the last used port is consistent with pod_k. Naturally, no port or rotation can be used beyond the destination port (constraints (F.5)–(F.7)). Moreover, a commodity cannot transit several times through the same port or the same rotation, what is ensured by constraints (F.8) and (F.9). To facilitate the expression of some constraints about the path followed by the commodities, we introduce Boolean variables $from^r_{k,p}$ (resp. $to^r_{k,p}$) which are true if commodity k enters (resp. leaves) the rotation r at port p, false otherwise. These variables are directly related to the previous ones as seen in constraints (F.10) and (F.11). These variables are also used to link the cargo flow to the definition of routes. Indeed, if commodity k is (un)loaded at a port p for a rotation r, it implies that the port p is used in this rotation (constraints (F.12) and (F.13)). Conversely, if a port p is used in rotation r, then there is at least one commodity that is (un)loaded in that port for that rotation (see constraints (F.14) and (F.15)). Finally, if a rotation is not used, no commodities can transit through it, and vice versa (constraint (F.16)). Note that this constraint is redundant, but in most cases, it allows finding some conflicts earlier.

3.4 Properties of Rotations and Vessels

In some MIP models in the literature (e.g. [18, 9]), each service is associated with a predefined type of vessel. While this choice facilitates the consideration of the specificities of each type of vessel, it leads to handling many rotations, few of which will be used in the end. Moreover,

$$\alpha_k = 1 \iff nbr_k > 0 \qquad k \in \mathcal{K} \tag{F.1}$$
$$\alpha_k = 1 \iff port_{k,1} = pol(k) \qquad k \in \mathcal{K} \tag{F.2}$$
$$\alpha_k = 1 \iff pod_k = pod(k) \qquad k \in \mathcal{K} \tag{F.3}$$
$$\texttt{Elt}(\{port_{k,i+1} \mid i \in [1, ts_{max}]\}, nbr_k) = pod_k \qquad k \in \mathcal{K} \tag{F.4}$$
$$i \geq nbr_k \iff rot_{k,i} = -1 \qquad k \in \mathcal{K}, i \in \mathcal{I} \tag{F.5}$$
$$nbr_k < i \Rightarrow port_{k,i} = -1 \qquad k \in \mathcal{K}, i \in \mathcal{I}^+ \tag{F.6}$$
$$port_{k,i} = -1 \Rightarrow nbr_k \leq i \qquad k \in \mathcal{K}, i \in \mathcal{I}^+ \tag{F.7}$$
$$\texttt{Alldiff-except}(\{port_{k,i} \mid i \in \mathcal{I}^+\}, -1) \qquad k \in \mathcal{K} \tag{F.8}$$
$$\texttt{Alldiff-except}(\{rot_{k,i} \mid i \in \mathcal{I}\}, -1) \qquad k \in \mathcal{K} \tag{F.9}$$
$$from^r_{k,p} = \sum_{i \in \mathcal{I}} (port_{k,i} = p) \cdot (rot_{k,i} = r) \qquad k \in \mathcal{K}, p \in \mathcal{P}, r \in \mathcal{R} \tag{F.10}$$
$$to^r_{k,p} = \sum_{i \in \mathcal{I}} (port_{k,i+1} = p) \cdot (rot_{k,i} = r) \qquad k \in \mathcal{K}, p \in \mathcal{P}, r \in \mathcal{R} \tag{F.11}$$
$$from^r_{k,p} = 1 \Rightarrow s^r_p \neq p \qquad k \in \mathcal{K}, p \in \mathcal{P}, r \in \mathcal{R} \tag{F.12}$$
$$to^r_{k,p} = 1 \Rightarrow s^r_p \neq p \qquad k \in \mathcal{K}, p \in \mathcal{P}, r \in \mathcal{R} \tag{F.13}$$
$$s^r_p \neq p \Rightarrow \texttt{Count}(\{port_{k,i} \mid k \in \mathcal{K}, i \in \mathcal{I}^+\}, \{p\}) \geq 1 \qquad p \in \mathcal{P}, r \in \mathcal{R} \tag{F.14}$$
$$s^r_p \neq p \Rightarrow \texttt{Count}(\{rot_{k,i} \mid k \in \mathcal{K}, i \in \mathcal{I}\}, \{r\}) \geq 1 \qquad p \in \mathcal{P}, r \in \mathcal{R} \tag{F.15}$$
$$v_r = 0 \iff \texttt{Count}(\{rot_{k,i} \mid k \in \mathcal{K}, i \in \mathcal{I}^+\}, \{r\}) = 0 \qquad r \in \mathcal{R} \tag{F.16}$$

Figure 5 Constraints related to cargo flow.

trying a new vessel type for a rotation requires the solver to change the assignment of a lot of variables. In our model, we choose to let the solver decide on the type of vessels associated with each rotation. Therefore, it is necessary to ensure that the type of vessels chosen for a rotation matches the features of the rotation. This requires the introduction of a certain number of variables whose values will then be fixed using `Elt` constraints. The variable κ_r represents the maximum capacity (expressed in TEUs) of commodities that can be transported via rotation r. Its value is 0 if the rotation is not used, the capacity of the type of vessel used otherwise. The variables ν^r_{min} and ν^r_{max} specify the minimum and maximum speeds of the rotation r (0 if the rotation is not used). The variable fp_r expresses the price per ton of fuel for rotation r (0 if the rotation is not operated). For each rotation r, we post the constraints (P.1)–(P.4) (see Figure 6). Likewise, some information (call costs, waiting time, ...) related to ports or canals also depends on the type of vessels associated with rotation r. For each rotation r, we then introduce the variables $wt_{p,r}$, $man^{in}_{p,r}$ and $man^{out}_{p,r}$ which specify respectively the waiting time of port/canal p and the maneuvring time to enter and leave port p. The cost of calling at port p for rotation r is represented by the variable $pc_{p,r}$ while the productivity for port p and rotation r is expressed by the variable $prod_{p,r}$. Constraints (P.5)–(P.9) ensure the consistency of these features. Finally, we consider the variable tc_r which, for each rotation r, specifies the daily cost of using the vessels associated with the rotation and its related constraint (P.10).

$$\texttt{Elt}(\{0\} \cup \{\kappa(v) \mid v \in \mathcal{V}\}, v_r) = \kappa_r \qquad r \in \mathcal{R} \qquad \text{(P.1)}$$
$$\texttt{Elt}(\{0\} \cup \{\nu_{min}(v) \mid v \in \mathcal{V}\}, v_r) = \nu^r_{min} \qquad r \in \mathcal{R} \qquad \text{(P.2)}$$
$$\texttt{Elt}(\{0\} \cup \{\nu_{max}(v) \mid v \in \mathcal{V}\}, v_r) = \nu^r_{max} \qquad r \in \mathcal{R} \qquad \text{(P.3)}$$
$$\texttt{Elt}(\{0\} \cup \{fp(fuel(v)) \mid v \in \mathcal{V}\}, v_r) = fp_r \qquad r \in \mathcal{R} \qquad \text{(P.4)}$$
$$\texttt{Elt}(\{0\} \cup \{wt(p, v) \mid v \in \mathcal{V}\}, v_r) = wt_{p,r} \qquad r \in \mathcal{R}, p \in \mathcal{P} \cup \mathcal{C} \qquad \text{(P.5)}$$
$$\texttt{Elt}(\{0\} \cup \{man^{in}(p, v) \mid v \in \mathcal{V}\}, v_r) = man^{in}_{p,r} \qquad r \in \mathcal{R}, p \in \mathcal{P} \cup \mathcal{C} \qquad \text{(P.6)}$$
$$\texttt{Elt}(\{0\} \cup \{man^{out}(p, v) \mid v \in \mathcal{V}\}, v_r) = man^{out}_{p,r} \qquad r \in \mathcal{R}, p \in \mathcal{P} \cup \mathcal{C} \qquad \text{(P.7)}$$
$$\texttt{Elt}(\{0\} \cup \{pc(p, v) \mid v \in \mathcal{V}\}, v_r) = pc_{p,r} \qquad r \in \mathcal{R}, p \in \mathcal{P} \cup \mathcal{C} \qquad \text{(P.8)}$$
$$\texttt{Elt}(\{1\} \cup \{prod(p, v) \mid v \in \mathcal{V}\}, v_r) = prod_{p,r} \qquad r \in \mathcal{R}, p \in \mathcal{P} \cup \mathcal{C} \qquad \text{(P.9)}$$
$$\texttt{Elt}(\{0\} \cup \{tc(v) \mid v \in \mathcal{V}\}, v_r) = tc_r \qquad r \in \mathcal{R} \qquad \text{(P.10)}$$

Figure 6 Constraint related to properties of rotations and vessels.

$$from^r_{k,p} = 1 \Rightarrow leave^r_{k,p} = 1 \qquad k \in \mathcal{K}, r \in \mathcal{R}, p \in \mathcal{P} \qquad \text{(L.1)}$$
$$to^r_{k,p} = 1 \Rightarrow leave^r_{k,p} = 0 \qquad k \in \mathcal{K}, r \in \mathcal{R}, p \in \mathcal{P} \qquad \text{(L.2)}$$
$$(s^r_p = p' \wedge from^r_{k,p'} = 0 \wedge to^r_{k,p'} = 0) \Rightarrow leave^r_{k,p} = leave^r_{k,p'} \qquad k \in \mathcal{K}, r \in \mathcal{R}, p, p' \in \mathcal{P} \qquad \text{(L.3)}$$
$$s^r_p = p \Rightarrow leave^r_{k,p} = 0 \qquad k \in \mathcal{K}, r \in \mathcal{R}, p \in \mathcal{P} \qquad \text{(L.4)}$$
$$\sum_{k \in \mathcal{K}} q(k) \cdot leave^r_{k,p} \leq \kappa_r \qquad r \in \mathcal{R}, p \in \mathcal{P} \qquad \text{(L.5)}$$

Figure 7 Constraints related to loads.

3.5 Vessel Load

We need to ensure that vessels do not leave each port loaded beyond their maximum capacity. This requires knowing, for each rotation, which commodities it carries at the exit of each port. To do this, we use a Boolean variable $leave^r_{k,p}$ per commodity k, port p, and rotation r. This variable is true if commodity k leaves port p via rotation r, false otherwise. Constraints (L.1) and (L.2) (see Figure 7) deal with the cases when the commodities are respectively loaded in rotation r and unloaded from rotation r while constraint (L.3) guarantees the transitivity all along the trip. Finally, the constraint (L.4) corresponds to the case where a port p does not appear in rotation r. Constraint (L.5) then allows ensuring that, for each port p, the load, when leaving the port, does not exceed the maximum capacity κ_r of rotation r.

3.6 Timestamp and Transit Times

3.6.1 Duration of Port Operations and Canal Traversal

To express the duration of loading/unloading operations in a port or the duration of traversing a canal, we introduce a variable t^r_p per port/canal p and rotation r. In the case of a canal the value of this variable is defined as equal to the duration of the traversal if the canal is used, 0 otherwise (see constraint (T.1) in Figure 8). For a port p, two cases are possible. If the port

$$
\begin{array}{lll}
t_p^r = trav(p) \cdot (s_p^r \neq p) & r \in \mathcal{R}, p \in \mathcal{C} & \text{(T.1)}\\
s_p^r = p \iff t_p^r = 0 & r \in \mathcal{R}, p \in \mathcal{P} & \text{(T.2)}\\
teu_{p,r} = \sum_{k \in \mathcal{K}} (from_{k,p}^r + to_{k,p}^r) \cdot q(k) & r \in \mathcal{R}, p \in \mathcal{P} & \text{(T.3)}\\
s_p^r \neq p \Rightarrow t_p^r = \left\lceil \frac{\mu.teu_{p,r}}{prod_{p,r}} \right\rceil & r \in \mathcal{R}, p \in \mathcal{P} & \text{(T.4)}\\
v_r = 0 \iff dep_r = -1 & r \in \mathcal{R} & \text{(T.5)}\\
v_r > 0 \Rightarrow \texttt{Maximum}(\{p.(s_p^r \neq p) \mid p \in \mathcal{P}\}) = dep_r & r \in \mathcal{R} & \text{(T.6)}\\
dep_r = p \Rightarrow time_{p,r}^{in} = 0 & r \in \mathcal{R}, p \in \mathcal{P} & \text{(T.7)}\\
time_{p,r}^{out} = time_{p,r}^{in} + t_p^r & r \in \mathcal{R}, p \in \mathcal{P} \cup \mathcal{C} & \text{(T.8)}\\
s_p^r = p' \Rightarrow st_p^r = \left\lceil \frac{\delta(p,p')}{\nu_p^r} \right\rceil & r \in \mathcal{R}, p, p' \in \mathcal{P} \cup \mathcal{C} & \text{(T.9)}\\
(v_r > 0 \land s_p^r \neq p) \Rightarrow \nu_p^r \geq \nu_{min}^r & r \in \mathcal{R}, p \in \mathcal{P} \cup \mathcal{C} & \text{(T.10)}\\
v_r > 0 \Rightarrow \nu_p^r \leq \nu_{max}^r & r \in \mathcal{R}, p \in \mathcal{P} \cup \mathcal{C} & \text{(T.11)}\\
s_p^r = p \iff st_p^r = 0 & r \in \mathcal{R}, p \in \mathcal{P} \cup \mathcal{C} & \text{(T.12)}\\
s_p^r = p \Rightarrow time_{p,r}^{in} = 0 & r \in \mathcal{R}, p \in \mathcal{P} \cup \mathcal{C} & \text{(T.13)}\\
s_p^r = p \iff \nu_p^r = 0 & r \in \mathcal{R}, p \in \mathcal{P} \cup \mathcal{C} & \text{(T.14)}\\
(s_p^r = p' \land p' \neq dep_r) & & \\
\Rightarrow time_{p',r}^{in} = time_{p,r}^{out} + man_{p,r}^{out} + st_p^r + wt_{p',r} + man_{p',r}^{in} & r \in \mathcal{R}, p, p' \in \mathcal{P} \cup \mathcal{C} & \text{(T.15)}\\
(from_{k,p}^r \land to_{k,p'}^r) & & \\
\Rightarrow (time_{p,r}^{in} < time_{p',r}^{in} \lor time_{p',r}^{in} < time_{p,r}^{in} < time_{p',r}^{in} + T_r) & r \in \mathcal{R}, p, p' \in \mathcal{P} \cup \mathcal{C}, k \in \mathcal{K} & \text{(T.16)}
\end{array}
$$

Figure 8 Constraints related to timestamps.

is not used in the rotation r, the variable t_p^r is 0 (see constraint (T.2)). Otherwise, its value depends on the number of TEUs loaded and unloaded in the port p for the rotation r. So, we consider the variable $teu_{p,r}$ which indicates the number of TEUs loaded and unloaded at port p for rotation r. Its value can be defined thanks to constraint (T.3). We can express the duration of the operations thanks to constraint (T.4). A crane movement allows moving a container whatever its size. To take into account the existence of 20-foot and 40-foot containers among the commodities to be handled, the parameter μ makes it possible to calculate the number of containers to be handled and thus the number of crane movements necessary from the number of containers expressed in TEUs.

3.6.2 Call Timestamps

In order to establish the schedule for each call, it is necessary to designate a port as the departure port in each rotation. To do this, we consider a variable dep_r per rotation r which has, for value, a port p if the rotation is used, -1 otherwise. The choice of the starting port being purely arbitrary, we choose the one with the largest index. This can be achieved thanks to constraints (T.5) and (T.6). Then, in our model, we consider two key moments: the moment when the vessel arrives at the berth (resp. enters the canal) and the moment when it leaves the berth (resp. leaves the canal). For each rotation r and each port/canal p, these two moments are represented respectively by the variables $time_{p,r}^{in}$ and $time_{p,r}^{out}$ which take their values in $[0, h_{max}]$. For each rotation, we consider that time 0 coincides with the time of arrival at the berth in the departure port thanks to constraint (T.7). The time of leaving a port or a canal depends only on the time of arrival and the duration of the operations in the port or the traversal of the canal (constraint (T.8)). Then, to determine the arrival time at a port or canal as a function of the exit time from the port/canal ahead of it in the rotation, we need to define the travel time as a function of the vessel's speed. The variables st_p^r and ν_p^r represent respectively the travel time from the port/canal p to its successor (if any) in rotation r and the speed (expressed in knots) used on this leg. The two variables are correlated by constraint (T.9). Of course, the speeds used must be consistent with the capabilities of the vessels operating the rotation (constraints (T.10) and (T.11). We can now define the time of arrival at the port/canal p' from its predecessor p in rotation r thanks to constraint (T.15). Note that for canals, we consider that the variables $man_{p,r}^{in}$ and $man_{p,r}^{in}$ are 0. This allows us to avoid defining the previous constraint according to the different possibilities of port/canal successions. In the case where a port p is not operated in a rotation r, we set the values of the variables st_p^r, ν_p^r and $time_{p,r}^{in}$ to the value 0 (constraints (T.12)–(T.14)).

Finally, if a commodity k is loaded in rotation r at port p and unloaded at port p', this imposes that the arrival at port p takes place before the arrival at port p' if the trip between p and p' does not pass through the departure port of rotation r. If this path passes through the departure port, then the arrival at port p' will occur in the next rotation, and the arrival at port p is between the two visits to port p'. For example, if we consider the rotation in Figure 2 (which lasts 840 hours) and a commodity sent from port p_2 to port p_4, a vessel operating this rotation enters port p_2 at hour 50 (in blue) and arrives in port p_4 at hour 600. In this case, we have $time_{p_2,r}^{in} < time_{p_4,r}^{in}$. On the other hand, if we consider a commodity going from p_4 to p_3, the vessel enters port p_3 at hour 275 before visiting p_4. This commodity will then be delivered only at the next passage of the vessel at time 1,115. We then have $time_{p_3,r}^{in} < time_{p_4,r}^{in} < time_{p_3,r}^{in} + T_r$. This is ensured by constraint (T.16).

3.6.3 Transit Times

To accurately consider the transit time of commodities, we need to know the key moments in their transportation, namely when they are loaded on board a rotation or unloaded. To simplify the model, we consider that a commodity is loaded on board a rotation when the rotation leaves the port and that it is unloaded when the rotation arrives at the port. These two times are represented by the variables $ctime_{i,k}^{in}$ and $ctime_{i,k}^{out}$ respectively. Constraints (T.17) and (T.18) (see Figure 9) ensure the correspondence between the key times of the rotations and the ones of the commodities.

The time spent by the commodity k in its ith rotation is represented by the variable $\delta_{i,k}$. It corresponds naturally to the difference between the exit time and the entry time. However, we must take into account the particular case where the journey passes through the port of departure. In this case, the commodities are unloaded at the next rotation. For example, if we consider the previous example, a commodity sent from port p_2 to port p_4 leaves port p_2 at hour 60 (in red) and arrives in port p_4 at hour 600 (in blue). This gives a travel time of 540 hours. On the other hand, a commodity shipped from port p_4 to port p_3 leaves port p_4 at time 620 and arrives at port p_3 at time 1,115 and thus takes 495 hours to reach its destination. Constraints (T.19) and (T.20) deal respectively with the first and second cases.

In the case where a transshipment takes place, the time that the commodities spend on the quay between the two rotations must be taken into account. Given the weekly frequency of the rotations, this time can be of the order of a week at most. To consider it more precisely, we introduce a variable $\Delta_{i,k}$ per commodity k and ith rotation used. The value of this variable is related to the weekly frequency of rotations. For example, consider a commodity

$$\texttt{Elt}_\texttt{m}(\{time^{out}_{p,r} \mid p \in \mathcal{P}, r \in R\}, port_{k,i}, rot_{k,i}) = ctime^{in}_{i,k} \qquad k \in \mathcal{K}, i \in \mathcal{I} \qquad \text{(T.17)}$$

$$\texttt{Elt}_\texttt{m}(\{time^{in}_{p,r} \mid p \in \mathcal{P}, r \in R\}, port_{k,i+1}, rot_{k,i}) = ctime^{out}_{i,k} \qquad k \in \mathcal{K}, i \in \mathcal{I} \qquad \text{(T.18)}$$

$$ctime^{in}_{i,k} \leq ctime^{out}_{i,k} \Rightarrow \delta_{i,k} = ctime^{out}_{i,k} - ctime^{in}_{i,k} \qquad k \in \mathcal{K}, i \in \mathcal{I}^+ \qquad \text{(T.19)}$$

$$(rot_{k,i} = r \wedge ctime^{in}_{i,k} > ctime^{out}_{i,k}) \Rightarrow \delta_{i,k} = ctime^{out}_{i,k} - ctime^{in}_{i,k} + T_r \qquad k \in \mathcal{K}, i \in \mathcal{I}^+ \qquad \text{(T.20)}$$

$$i + 1 < nbr_k \Rightarrow \Delta_{i,k} = (ctime^{in}_{i+1,k} - ctime^{out}_{i,k})\%168 \qquad k \in \mathcal{K}, i \in \mathcal{I} \qquad \text{(T.21)}$$

$$i + 1 \geq nbr_k \Rightarrow \Delta_{i,k} = 0 \qquad k \in \mathcal{K}, i \in \mathcal{I} \qquad \text{(T.22)}$$

$$\sum_{i \in \mathcal{I}^+} \delta_{i,k} + \sum_{i \in \mathcal{I}} \Delta_{i,k} \leq tt_{max}(k) \qquad k \in \mathcal{K} \qquad \text{(T.23)}$$

Figure 9 Constraints related to transit times.

k that arrives at a port p at hour 200 (according to its $ctime^{out}_{i,k}$ value) on a rotation r and leaves it at hour 2,000 (according to its $ctime^{in}_{i+1,k}$ value) via a rotation r'. The weekly frequency of the rotations r and r' implies that, in practice, commodity k leaves the port at hour 320. Indeed, a vessel of rotation r' leaves the port at hours 152 (i.e. $2,000$ modulo (7×24)), 320, 488, …If the arrival in the port is later than the departure from the port (according to the values $ctime^{out}_{i,k}$ and $ctime^{in}_{i+1,k}$), it means that the commodity must wait for the vessel of the next week and so we have to add 168 hours to the considered difference. Constraint (T.21) takes into account these two cases. The ith rotations that are not used for commodity k have a $\Delta_{i,k}$ variable whose value is zero (constraint (T.22)). The transit time of commodity k can then be guaranteed by constraint (T.23). Note that not all commodities have a transit time constraint. Also, if commodity k does not have a maximum transit time constraint, the variables and constraints presented here are not considered for it.

3.7 Vessel Availability

Since the frequency of the rotations is weekly, each port is visited by one vessel operating the rotation each week. The number of vessels needed is therefore the total time of the rotation divided by the duration of one week. If the variables n_r and T_r represent respectively the number of vessels used by the rotation r and the total time of the rotation r, we can impose constraint (A.1) (see Figure 10). The time of the rotation is, of course, zero if the rotation is not used (constraint (A.2)). Otherwise, since each rotation starts at time 0, the total time is the arrival time at the departure port from the last port of the rotation (constraint (A.3)). Finally, we guarantee that, for each type of vessel, the number of vessels used does not exceed the number of available vessels thanks to constraint (A.4).

3.8 Objective Function

Briefly, the objective function is the difference between the revenues generated by accepting commodities into the network and the total costs of transporting them (fuel, vessel operations, port calls, …). The fuel cost depends on the fuel price and the fuel consumption of each trip made. For this latter, we consider a variable $cons^r_p$ per port and rotation that specifies the amount of fuel consumed per hour by the rotation r for the trip made between the port/canal p and its successor. In the absence of successors, the variable $cons^r_p$ has, of course, the value

$$n_r = \left\lceil \frac{T_r}{7 \times 24} \right\rceil \qquad r \in \mathcal{R} \qquad \text{(A.1)}$$

$$v_r = 0 \iff T_r = 0 \qquad r \in \mathcal{R} \qquad \text{(A.2)}$$

$$dep_r = p' \wedge s_p^r = p' \Rightarrow T_r = time_{p,r}^{out} + man_{p,r}^{out} + st_p^r + wt_{p',r} + man_{p',r}^{in} \qquad r \in \mathcal{R}, p, p' \in \mathcal{P} \qquad \text{(A.3)}$$

$$\sum_{r \in R} n_r.(v_r = v) \leq nb(v) \qquad v \in \mathcal{V} \qquad \text{(A.4)}$$

Figure 10 Constraints related to vessel availability.

$$\texttt{Elt}_{\texttt{m}}(\{cons(v,\nu) \mid v \in \{0\} \cup \mathcal{V}, \nu \in \{0\} \cup [\nu_{min}(v), \nu_{max}(v)]\}, v_r, \nu_p^r) = cons_p^r \qquad r \in \mathcal{R}, p \in \mathcal{P} \cup \mathcal{C} \qquad \text{(O.1)}$$

$$teu_p^{ts} = \sum_{k \in \mathcal{K} | p \neq pod(k)} to_{k,p}^r \cdot q(k) \qquad p \in \mathcal{P} \qquad \text{(O.2)}$$

Figure 11 Constraints related to the objective function.

0. The quantity consumed here depends only on the type of vessel used and the speed. In constraint (O.1) (see Figure 11), we assume that $cons(v, \nu)$ is 0 if v or ν is 0.

The costs associated with transshipment depend on the port and the quantity of commodities transshipped. So we need to represent the quantity of commodities transshipped at each port. To do this, we introduce a variable teu_p^{ts} per port. The commodities k transshipped at port p are those that are unloaded at port p (i.e., those for which $to_{k,p}^r$ is 1) and for which port p is not their destination port (see constraint (O.2)). We can now express our objective function based on revenues (R), fuel costs (C), canal and port call costs (E), vessel operating costs (X), and transshipment costs (T):

$$\begin{aligned}
\max \quad & \sum_{k \in \mathcal{K}} rev(k) \cdot q(k) \cdot \alpha_k & (R)\\
- \quad & \sum_{r \in R} \sum_{p \in \mathcal{P} \cup \mathcal{C}} fp_r \cdot cons_p^r \cdot st_p^r & (C)\\
- \quad & \sum_{r \in R} \sum_{p \in \mathcal{P} \cup \mathcal{C}} pc_{pr} \cdot (s_p^r \neq p) & (E)\\
- \quad & 7 \sum_{r \in R} tc_r \cdot n_r & (X)\\
- \quad & \sum_{p \in \mathcal{P}} ts(p) \cdot teu_p^{ts} & (T)
\end{aligned}$$

3.9 Additional Constraints

Given the size of the search space, it may be desirable to avoid certain symmetries as much as possible. Starting each rotation at time 0 (see constraint (T.7)) allows for avoiding any translation on the time axis. However, other symmetries may exist. For example, the rotations are interchangeable. To avoid this, we can ensure that the first rotations are used in priority and these rotations are sorted in decreasing order of their duration thanks to constraint (S.1) (see Figure 12). One of the main practical difficulties of the LSNDP problem lies in the enumeration of the different possible circuits. For a circuit of length

$$T_1 \geq T_2 \geq \ldots \geq T_r \tag{S.1}$$

$$\sum_{r \in \mathcal{R}} (s_p^r \neq p) \leq \max(|\mathcal{K}_p^{pol}|, |\mathcal{K}_p^{pod}|) \qquad p \in \mathcal{P}_2 \tag{S.2}$$

$$v_r = v \Rightarrow s_p^r = p \qquad r \in \mathcal{R}, p \in \mathcal{P}_3 \tag{S.3}$$

$$teu_p^{ts} = 0 \qquad p \in \mathcal{P}_4 \tag{S.4}$$

Figure 12 Possible additional constraints where $\mathcal{P}_i$ denotes the set of ports impacted by the constraint (S.i).

ℓ, since it is possible to go from one port to any other, the solver may have to consider a non-negligible part of the $\ell!$ possible permutations. Since, in addition, several rotations are usually considered simultaneously, this can quickly become very time-consuming. To reduce the number of rotations and thus of circuits to consider, we introduce constraint (S.2). Let $\mathcal{K}_p^{pol}$ and $\mathcal{K}_p^{pod}$ denote respectively the set of commodities for which p is the origin port and one for which p is the destination port. This constraint ensures that the number of rotations that uses a port p does not exceed the maximum between $|\mathcal{K}_p^{pol}|$ and $|\mathcal{K}_p^{pod}|$. In a way, it eliminates some solutions in which the call in the port would only be used for transshipments. Generally, calling a port for only transshipments is not wished by shipping companies, except for some particular ports (e.g. hubs). Thanks to the flexibility of CP, we can add this constraint depending on the needs of the shipowner.

On the other hand, some ports cannot handle certain types of vessels. For example, the port of Dutch Harbor in Alaska is not deep enough. It can therefore only handle small container ships. Thus, if v vessels cannot berth at port p, we impose constraint (S.3) for each rotation r. Similarly, some ports do not have enough space to store containers. It is therefore impossible to carry out transshipments there. For such ports, we can then exploit constraint (S.4) to prohibit any transshipment.

4 Experiments

4.1 Experimental Protocol and Implementation Details

The LINER-LIB benchmark[1] [7] is the reference for experiments on the LSNDP problem. It consists of seven instances with 12 to 197 ports, thus allowing the evaluation of both complete and incomplete methods. In order to have instances of a reasonable and varied size, we have produced sub-instances from instances of the LINER-LIB benchmark. To do this, from an instance, we select n ports in the following way. The first selected port is the one that handles the most commodities. The next $n-1$ ports are the ones that exchange the most commodities with the already selected ports. For our test set, we considered the smallest instance (`Baltic`) of the LINER-LIB set and 40 instances produced from the instances `Baltic`, `EuropeAsia`, `Mediterranean` and `WAF`. The number of ports varies from 3 to respectively 11, 10, 10 and 17. Moreover, as the LINER-LIB benchmark does not take into account productivity, waiting or maneuvring times, we generate randomly these values. Note that this partial random generation introduces no bias, since these values have a negligible impact on the solving efficiency. For the following experiments, the number of

[1] https://github.com/blof/LINERLIB/

rotations r_{max} is fixed at 4 and that of transhipments at 1 per commodity (a higher value not being desired by the experts) while the maximum duration h_{max} is set to 12 weeks. The chosen values of r_{max} and h_{max} seem reasonable to us taking into account the commodities to be transported and the distances to be covered. In practice, the optimal solutions we found require less time and fewer rotations than our choice of parameters allows. For μ, we take the value 0.54 given by the experts. The instances we consider are available at `https://pageperso.lis-lab.fr/cyril.terrioux/LSNDP/instances.zip`.

The presented model is implemented in the OR-Tools CP-SAT solver (version 9.6.2534 [26]) via its Python interface. This choice is first guided by the solver efficiency, since the OR-Tools CP-SAT solver won several gold medals during the past MiniZinc Challenges [25]. Moreover, lazy clause generation [20] provides good results for the LSFRP problem [14]. Finally, another advantage is the possibility of exploiting a certain form of parallelism. Hence, for the solving, we run from 1 to 16 threads. When a single thread is run, it corresponds to the CP-SAT solver. Except for the number of threads, all the parameters are the default ones. The experiments are being conducted on servers with Intel Xeon Gold 5218R processors running at 2.1 GHz and 192 GB of memory with a time limit of two hours. When exploiting several threads, each instance is solved 10 times and the reported runtime is the average time. The solving step involving t threads is denoted $\times t$. We apply it to our model M, but also, to two derived versions denoted M-1,2 and M-2. Model M-1,2 does not consider constraints (S.1) and (S.2) while model M-2 uses constraints (S.1), but not constraints (S.2).

4.2 Results

First, we compare our model with its two derived versions from the efficiency viewpoint (see Table 1[2] and Figure 13). Clearly, model M is the most efficient. Indeed, the addition of constraints (S.1) and (S.2) allows us to solve optimally more instances (24 instances for model M against 14 for models M-1,2 and M-2 when using a single thread) while reducing significantly the runtime. As there often exists an arc between each pair of ports/canals, the circuit constraint admits a huge number of allowed tuples. In practice, the number of allowed tuples studied by the solver is mainly restricted by the load and transit time constraints or the objective function. In the latter case, as the objective function considers all the rotations, it may take some time for the solver to realize that a rotation is not suitable. So finding an optimal solution may require exploring a huge number of feasible solutions. Constraints (S.1) and (S.2) allow us to reduce this number significantly, as we can see in our results. Then, as shown in Table 1, exploiting several threads allows improving the efficiency, but mostly by reducing the runtime. For instance, using 16 threads instead of a single one leads to reducing the runtime by a factor of 5 on average and up to 20 at best.

Figure 13 indicates that the runtime increases exponentially with the number of ports. However, other parameters affect the runtime like the number of commodities to be processed or the type of instances. For example, our model performs well on Baltic and WAF instances, which corresponds to feedering instances (i.e. a collection of small services that ensures the transport of commodities between some main ports and satellite ones). In contrast, it turns out to be less efficient for instances like EuropeAsia ones that connect the more important ports of two commercial areas. Then, our model finds interesting solutions even if it does not accept all the commodities or visit all the ports (see Table 2). One explanation is related to

[2] See Appendix A for the other instances.

Table 1 Runtimes in seconds for some representative instances (a runtime different from 7,200 s corresponds to an instance solved optimally).

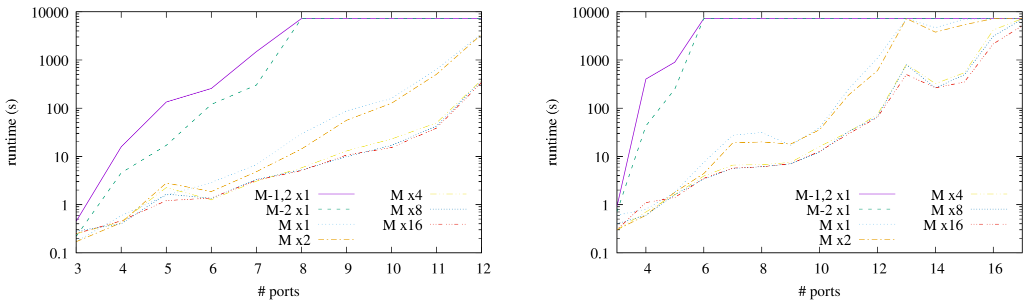
Instance	M-1,2 ×1	M-2 ×1	M×1	M×2	M×4	M×8	M×16
Baltic	7,200	7,200	3,502	3,385	386	372	343
Baltic_sub7	1,483	301	6.7	4.8	3.0	3.3	3.2
EuropeAsia_sub7	7,200	7,200	7,200	7,200	7,200	7,200	7,200
Mediterranean_sub7	7,200	7,200	7,200	7,200	7,200	7,001	6,234
WAF_sub7	7,200	7,200	27.4	19.0	6.6	5.6	5.7
WAF_sub17	7,200	7,200	7,200	7,200	7,200	7,021	5,117

Table 2 Information of some instances and solutions (the value in k$ of the best solution found, the number of visited ports, accepted commodities and used rotations).

Instance			Solution			
Name	$\|\mathcal{P}\|$	$\|\mathcal{K}\|$	Cost	#ports	#comm.	#rot
Baltic	12	22	4,752	10	16	3
Baltic_sub7	7	12	2,508	6	9	2
EuropeAsia_sub7	7	42	4,228	6	23	3
Mediterranean_sub7	7	26	225	6	15	2
WAF_sub7	7	12	5,823	7	12	3
WAF_sub17	17	32	11,952	10	16	4

the way the LINER-LIB benchmark was built (namely by aggregating data from different shipowners without ensuring that the considered fleet can handle all the commodities).

Finally, our approach manages to optimally solve some instances of up to 17 ports. While the literature reports solving of up to a dozen ports, it is difficult to compare accurately with existing exact methods: implementations are generally not available and each work treats the problem with a different point of view, in particular regarding the working hypotheses or the cost function to optimize (see Section 5).



(a) (b)

Figure 13 Runtime (in s) for Baltic (a) and WAF (b) instances w.r.t. the number of ports.

5 Related Work

The Liner Shipping Fleet Repositioning Problem (LSFRP [28]) aims to adapt the network in order to take into account some evolution of the customer needs (e.g. seasonality, port congestion, increase or decrease of the demands, . . .). Moving container ships from one service to another while considering commodity transport is a complex and expensive task for shipping companies. In this context, the CP approach presented in [14] turns out to be more efficient than MIP ones. In the case of LSFRP, services are already defined while LSNDP aims to design them. Moreover, the number of vessels and commodities to handle may be reduced for LSFRP. Finally, an LSFRP instance corresponds to a one-shot task while an LSNDP one leads to a schedule for several weeks or months.

LSNDP is close to Vehicle Routing Problem (VRP [15]) and its variants like the pickup and delivery problem (PDP [11]). Indeed, in both cases, vehicles carry commodities from one location to another. The first difference is that generally, for VRP and PDP, a commodity is carried by a single vehicle whereas, for LSNDP, the transport can be achieved by several vessels operating different rotations thanks to transshipments. Taking into account transshipments makes the problem more difficult. This requires additional variables and constraints, while significantly increasing the number of possible routes for commodities (and therefore the number of feasible solutions to study). Moreover, in general, VRP and PDP aim to transport all the commodities while, for LSNDP, some commodities may be rejected. Finally, the objective function is often more complex for LSNDP than for VRP and PDP. For instance, the variety of considered costs (e.g. call cost, transshipment cost, fuel cost, charter rate, ...) is more important.

Regarding the exact solving of LSNDP, unlike our model, the models proposed in [18, 23, 22] use a constant speed for each leg and do not handle transit time constraints. Those of [18, 23] cannot reject a commodity. In contrast, [18] takes into account the empty container repositioning while [23, 22] consider the transshipment costs. The objective functions consist in minimizing costs [18, 23] or maximizing the profit [22]. Regarding the type of service, the three models consider a more general form than ours. For instance, they can exploit butterfly services (i.e. services that can call several times in the same port). However, such an extension could be taken into account in our model by duplicating ports as we do for canals and relaxing some constraints like constraints (F.9). Note that the experimentations achieved in [22] rely on the LINER-LIB benchmark, but the proposed approach does not succeed in solving optimally the Baltic instance.

6 Conclusions and Perspectives

In this paper, we have proposed a first CP model to solve the LSNDP problem. The first practical results are very encouraging with optimally solved instances with up to 17 ports and show the interest in a CP approach. In the future, this model will have to be extended to better handle some kinds of instances and take into account other forms of services (e.g. butterflies) or the constraints imposed by the International Maritime Organization (IMO) concerning the gas emissions of ships (e.g. related to carbon intensity indicator). Another extension would be to differentiate containers by type (full, empty, reefers, ...). In particular, the repositioning of empty containers is an important issue, while the transport of reefers is highly profitable and raises specific questions. Moreover, to facilitate scaling up, the use of incomplete methods will need to be explored more deeply.

References

1 R. Agarwal and Ö. Ergun. Ship Scheduling and Network Design for Cargo Routing in Liner Shipping. *Transportation Science*, 42(2):175–196, 2008. doi:10.1287/trsc.1070.0205.

2 J. F. Álvarez. Joint routing and deployment of a fleet of container vessels. *Maritime Economics & Logistics*, 11(2):186–208, 2009. doi:10.1057/mel.2009.5.

3 N. Beldiceanu, M. Carlsson, and J.-X. Rampon. Global constraint catalog. Technical Report T2012:03, TASC-SICS-LINA, 2014.

4 N. Beldiceanu and E. Contejean. Introducing global constraints in CHIP. *Mathematical and Computer Modelling*, 20(12):97–123, 1994.

5 B. D. Brouer, G. Desaulniers, C. V. Karsten, and D Pisinger. A Matheuristic for the Liner Shipping Network Design Problem with Transit Time Restrictions. In *Computational Logistics - 6th International Conference, ICCL*, volume 9335 of *Lecture Notes in Computer Science*, pages 195–208. Springer, 2015. doi:10.1007/978-3-319-24264-4_14.

6 B. D. Brouer, C. V. Karsten, and D. Pisinger. Optimization in liner shipping. *Annals of Operations Research*, 271(1):205–236, 2018. doi:10.1007/s10479-018-3023-8.

7 B. D. Brouer, J. F. Álvarez, C. E. M. Plum, D. Pisinger, and M. M. Sigurd. A base integer programming model and benchmark suite for liner-shipping network design. *Transportation Science*, 48(2):281–312, 2014. doi:10.1287/trsc.2013.0471.

8 M. Carlsson, G. Ottosson, and B. Carlson. An open-ended finite domain constraint solver. In *Programming Languages: Implementations, Logics, and Programs (PLILP)*, volume 1292 of *Lecture Notes in Computer Science*, pages 191–206, 1997. doi:10.1007/BFb0033845.

9 M. Christiansen, E. O. Hellsten, D. Pisinger, D. Sacramento, and C. Vilhelmsen. Liner shipping network design. *European Journal of Operational Research*, 286, 2020. doi:10.1016/j.ejor.2019.09.057.

10 R. Cymer. Dulmage-Mendelsohn Canonical Decomposition as a generic pruning technique. *Constraints*, 17(3):234–272, 2012.

11 Y. Dumas, J. Desrosiers, and F. Soumis. The pickup and delivery problem with time windows. *European Journal of Operational Research*, 54(1):7–22, 1991. doi:10.1016/0377-2217(91)90319-Q.

12 C. V. Karsten, B. D. Brouer, G. Desaulniers, and D. Pisinger. Time constrained liner shipping network design. *Transportation Research. Part E: Logistics and Transportation Review*, 105:152–162, 2017. doi:10.1016/j.tre.2016.03.010.

13 C. V. Karsten, D. Pisinger, S. Røpke, and B. D. Brouer. The time constrained multi-commodity network flow problem and its application to liner shipping network design. *Transportation Research Part E: Logistics and Transportation Review*, 76:122–138, 2015. doi:10.1016/j.tre.2015.01.005.

14 E. Kelareva, K. Tierney, and P. Kilby. CP methods for scheduling and routing with time-dependent task costs. *EURO Journal on Computational Optimization*, 2(3):147–194, 2014. doi:10.1007/s13675-014-0022-7.

15 P. Kilby and P. Shaw. Vehicle routing. In F. Rossi, P. van Beek, and T. Walsh, editors, *Handbook of Constraint Programming*, volume 2 of *Foundations of Artificial Intelligence*, pages 801–836. Elsevier, 2006. doi:10.1016/S1574-6526(06)80027-1.

16 D. Kizilay, P. Van Hentenryck, and D. Türsel Eliiyi. Constraint programming models for integrated container terminal operations. *European Journal of Operational Research*, 286(3):945–962, 2020. doi:10.1016/j.ejor.2020.04.025.

17 A. Krogsgaard, D. Pisinger, and J. Thorsen. A flow-first route-next heuristic for liner shipping network design. *Networks*, 72(3):358–381, 2018. doi:10.1002/net.21819.

18 Q. Meng and S. Wang. Liner shipping service network design with empty container repositioning. *Transportation Research Part E: Logistics and Transportation Review*, 47(5):695–708, 2011. doi:10.1016/j.tre.2011.02.004.

19 Q. Meng and S. Wang. Optimal operating strategy for a long-haul liner service route. *European Journal of Operational Research*, 215(1):105–114, 2011. doi:10.1016/j.ejor.2011.05.057.

20 O. Ohrimenko, P. J. Stuckey, and M. Codish. Propagation via lazy clause generation. *Constraints An International Journal*, 14(3):357–391, 2009. doi:10.1007/s10601-008-9064-x.

21 C. E. M. Plum, D. Pisinger, J. J. Salazar-González, and M. M. Sigurd. Single liner shipping service design. *Computers & Operations Research*, 45:1–6, 2014. doi:10.1016/j.cor.2013.11.018.

22 C. E. M. Plum, D. Pisinger, and M. M. Sigurd. A service flow model for the liner shipping network design problem. *European Journal of Operational Research*, 235(2):378–386, 2014. doi:10.1016/j.ejor.2013.10.057.

23 L. B. Reinhardt and D. Pisinger. A branch and cut algorithm for the container shipping network design problem. *Flexible Services and Manufacturing Journal*, 24(3):349–374, 2012. doi:10.1007/s10696-011-9105-4.

24 D. Sacramento, C. Solnon, and D. Pisinger. Constraint Programming and Local Search Heuristic: a Matheuristic Approach for Routing and Scheduling Feeder Vessels in Multi-terminal Ports. *Annals of Operations Research*, 1(4), 2020. doi:10.1007/s43069-020-00036-x.

25 P. J. Stuckey, T. Feydy, A. Schutt, G. Tack, and J. Fischer. The MiniZinc Challenge 2008-2013. *AI Magazine*, 35(2):55–60, 2014.

26 OR-Tools team. OR-Tools CP-SAT solver, 2023. URL: https://developers.google.com/optimization/cp/cp_solver.

27 K. Thun, H. Andersson, and M. Christiansen. Analyzing complex service structures in liner shipping network design. *Flexible Services and Manufacturing Journal*, 29(3-4):535–552, 2017. doi:10.1007/s10696-016-9262-6.

28 K. Tierney. *Optimizing Liner Shipping Fleet Repositioning Plans*, volume 57 of *Operations research / computer science interfaces series*. Springer, 2015. doi:10.1007/978-3-319-17665-9.

29 P. Van Hentenryck and J.-P. Carillon. Generality versus Specificity: An Experience with AI and OR Techniques. In *Proceedings of the National Conference on Artificial Intelligence*, pages 660–664, 1988.

30 S. Wang and Q. Meng. Liner shipping network design with deadlines. *Computers & Operations Research*, 41:140–149, 2014. doi:10.1016/j.cor.2013.08.014.

A Detailed Results

Tables 3 and 4 provide the same information as Tables 1 and 2 but for all the instances we consider.

Table 3 Runtimes in seconds for all the considered instances (a runtime different from 7,200 s corresponds to an instance solved optimally).

Instance	M-1,2 $\times$1	M-2 $\times$1	$M\times$1	$M\times$2	$M\times$4	$M\times$8	$M\times$16
Baltic_sub3	0.4	0.2	0.2	0.2	0.2	0.3	0.3
Baltic_sub4	15.8	4.5	0.6	0.4	0.4	0.4	0.5
Baltic_sub5	134	16.9	1.6	2.8	2.3	1.6	1.2
Baltic_sub6	256	120	2.9	1.9	1.3	1.4	1.4
Baltic_sub7	1,483	301	6.7	4.8	3.0	3.3	3.2
Baltic_sub8	7,200	7,200	29.2	14.2	5.8	5.2	5.1
Baltic_sub9	7,200	7,200	87.7	55.8	12.9	9.8	10.6
Baltic_sub10	7,200	7,200	158	126	22.9	16.9	15.2
Baltic_sub11	7,200	7,200	637	504	49.6	42.6	38.7
Baltic	7,200	7,200	3,502	3,385	386	372	343
EuropeAsia_sub3	2.5	1.0	0.9	1.0	1.2	1.3	1.2
EuropeAsia_sub4	157	244	134	100	37.3	18.1	10.5
EuropeAsia_sub5	7,200	7,200	7,200	3,736	4,101	4,459	2,356
EuropeAsia_sub6	7,200	7,200	7,200	7,200	7,200	7,200	7,200
EuropeAsia_sub7	7,200	7,200	7,200	7,200	7,200	7,200	7,200
EuropeAsia_sub8	7,200	7,200	7,200	7,200	7,200	7,200	7,200
EuropeAsia_sub9	7,200	7,200	7,200	7,200	7,200	7,200	7,200
EuropeAsia_sub10	7,200	7,200	7,200	7,200	7,200	7,200	7,200
Mediterranean_sub3	0.4	0.2	0.3	0.3	0.3	0.3	0.3
Mediterranean_sub4	2.8	0.8	0.8	0.8	0.8	0.9	1.0
Mediterranean_sub5	85.0	28.9	26.5	22.4	5.2	3.8	3.8
Mediterranean_sub6	621	229	281	307	30.6	25.2	21.5
Mediterranean_sub7	7,200	7,200	7,200	7,200	7,200	7,001	6,234
Mediterranean_sub8	7,200	7,200	7,200	7,200	7,200	7,200	7,200
Mediterranean_sub9	7,200	7,200	7,200	7,200	7,200	7,200	7,200
Mediterranean_sub10	7,200	7,200	7,200	7,200	7,200	7,200	7,200
WAF_sub3	0.9	0.7	0.6	0.3	0.3	0.4	0.3
WAF_sub4	401	42.8	0.8	0.6	0.7	0.6	1.1
WAF_sub5	896	248	1.8	1.8	1.6	1.6	1.4
WAF_sub6	7,200	7,200	7.5	4.4	3.9	3.4	3.5
WAF_sub7	7,200	7,200	27.4	19.0	6.6	5.6	5.7
WAF_sub8	7,200	7,200	31.3	19.9	6.7	6.1	6.1
WAF_sub9	7,200	7,200	17.0	18.0	7.5	7.0	7.0
WAF_sub10	7,200	7,200	39.4	35.7	16.0	12.3	12.7
WAF_sub11	7,200	7,200	249	187	32.2	32.6	29.3
WAF_sub12	7,200	7,200	1,093	600	73.4	66.7	64.9
WAF_sub13	7,200	7,200	7,200	7,200	815	792	499
WAF_sub14	7,200	7,200	4,662	3,779	323	262	263
WAF_sub15	7,200	7,200	7,034	5,365	542	494	349
WAF_sub16	7,200	7,200	7,200	7,200	4,182	3,106	2,154
WAF_sub17	7,200	7,200	7,200	7,200	7,200	7,021	5,117

Table 4 Some information about instances and solutions. For solutions, we provide the value (in k$) of the best solution found, the number of visited ports, one of accepted commodities and one of used rotations.

Instance			Solution			
Name	$\|\mathcal{P}\|$	$\|\mathcal{K}\|$	Cost	#ports	#comm.	#rot
Baltic_sub3	3	4	1,876	3	4	1
Baltic_sub4	4	6	1,895	3	4	1
Baltic_sub5	5	8	2,074	5	8	2
Baltic_sub6	6	10	2,074	5	8	2
Baltic_sub7	7	12	2,508	6	9	2
Baltic_sub8	8	14	3,322	7	10	3
Baltic_sub9	9	16	3,733	8	11	3
Baltic_sub10	10	18	4,187	9	13	3
Baltic_sub11	11	20	4,345	10	15	3
Baltic	12	22	4,752	10	16	3
EuropeAsia_sub3	3	6	616	3	4	1
EuropeAsia_sub4	4	12	616	3	4	1
EuropeAsia_sub5	5	20	1,463	4	8	1
EuropeAsia_sub6	6	30	1,463	4	8	1
EuropeAsia_sub7	7	42	4,228	6	23	3
EuropeAsia_sub8	8	56	4,425	6	19	2
EuropeAsia_sub9	9	72	4,425	6	19	2
EuropeAsia_sub10	10	89	2,935	7	24	2
Mediterranean_sub3	3	5	177	3	5	1
Mediterranean_sub4	4	9	177	3	5	1
Mediterranean_sub5	5	14	196	4	8	1
Mediterranean_sub6	6	20	196	4	8	1
Mediterranean_sub7	7	26	225	6	15	2
Mediterranean_sub8	8	34	302	8	30	2
Mediterranean_sub9	9	43	509	8	30	2
Mediterranean_sub10	10	53	605	9	38	2
WAF_sub3	3	4	1,293	3	4	1
WAF_sub4	4	6	1,308	4	6	3
WAF_sub5	5	8	2,329	5	8	3
WAF_sub6	6	10	2,911	6	10	3
WAF_sub7	7	12	5,823	7	12	3
WAF_sub8	8	14	5,823	7	12	3
WAF_sub9	9	16	5,823	7	12	3
WAF_sub10	10	18	7,543	8	14	3
WAF_sub11	11	20	8,831	9	15	4
WAF_sub12	12	22	9,764	10	16	4
WAF_sub13	13	24	11,282	11	18	4
WAF_sub14	14	26	11,952	10	16	4
WAF_sub15	15	28	11,952	10	16	4
WAF_sub16	16	30	11,952	10	16	4
WAF_sub17	17	32	11,952	10	16	4