Ocean Engineering 286 (2023) 115525

Contents lists available at ScienceDirect

Ocean Engineering

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

The pipelines and cable trays location problem in naval design

Víctor Blanco [a,b,1], Gabriel González [a,c,1], Yolanda Hinojosa [c,d,1], Diego Ponce [c,e,*,1], Miguel A. Pozo [c,e,1], Justo Puerto [c,e,1]

[a] Institute of Mathematics (IMAG), Universidad de Granada, Spain
[b] Dpt. Quant. Methods for Economics & Business, Universidad de Granada, Spain
[c] Institute of Mathematics (IMUS), Universidad de Sevilla, Spain
[d] Dpt. Applied Economics I, Universidad de Sevilla, Spain
[e] Dpt. Stats & OR, Universidad de Sevilla, Spain

ARTICLE INFO

MSC:
46N10
65K05
90B10
90C35

Keywords:
Pipeline routing
Cable trays location
Network design
Matheuristics
Naval engineering

ABSTRACT

This paper deals with the determination of optimal locations for pipelines and cable trays in naval design. The problem consists of finding the number and types of cable tray routes to be created between various devices in order to minimize a user defined cost function. We reduce the problem to an *ad hoc* min-cost multicommodity flow problem with additional constraints imposed by technical requirements. This problem is solved for small-sized instances by using off-the-shelf optimization solvers. We also develop an exact relax-and-cut strategy that allows to handle medium-sized instances. For larger instances, we propose a family of heuristic algorithms consisting on the combination of two phases: (I) Construction of initial cable trays paths; and (II) Transformation to feasible cable trays verifying the technical requirements. For each of them, we also propose different strategies which give rise to several algorithms. These algorithms are compared on a computational experience using two types of instances: the first one based on random instances of different sizes and the second one based on instances with well-defined corridors to asses the availability of our methodology to enforce the creation of cable trays. Finally, we also analyze a real size case study provided by our industrial partner, Ghenova, a leading Naval Engineering company, validating our proposal to find solutions for this problem.

1. Introduction

1.1. Motivation

Routing pipelines or cables *optimally* is a recurrent problem in naval design and the electrical industry since it is the most important activity during the detail-design phase. Actually, it takes over 50% of the total detail-design person-hours and all other activities of detail design depend on it (Park and Storch, 2002). Designers face the demanding task of strategically placing multiple elements while ensuring obstacle avoidance, providing adequate machinery space, and meeting technical requirements for compatibility and manufacturability. These objectives must be accomplished within a confined space, leaving little room for maneuvering.

Certain elements, such as water pipes and electrical circuits, require separation, while others, like power, phone, or optical fiber cables, can be consolidated using cable trays. These trays effectively reduce the spatial footprint by grouping elements of the same type together. Although software tools exist to aid engineers in visualizing pipe placement and identifying overlaps or interferences, achieving an optimal pipe and cable layout still heavily relies on experienced designers, who often command higher salaries, leading to increased expenses. By automating the design process for pipelines and cable trays, costs are reduced as the need for a specialized designer dedicated to this task is eliminated. This automation holds the potential for achieving a better, or even optimal, design in a more cost-effective manner. Additionally, it would significantly decrease the time required to complete this task and ultimately result in safer designs.

The **P**ipelines and **C**able **T**rays **L**ocation **P**roblem (PCTLP) consists of finding the overall number and types of pipeline and cable tray paths to be created between various devices and equipment in a reduced space (building, ship, etc.) under constructability constraints in order to minimize a user defined cost function. This function usually depends

* Corresponding author at: Institute of Mathematics (IMUS), Universidad de Sevilla, Spain.
E-mail addresses: vblanco@ugr.es (V. Blanco), ggdominguez@ugr.es (G. González), yhinojos@us.es (Y. Hinojosa), dponce@us.es (D. Ponce), miguelpozo@us.es (M.A. Pozo), puerto@us.es (J. Puerto).

[1] Evenly contributed to all aspect of the research and writing of the manuscript.

https://doi.org/10.1016/j.oceaneng.2023.115525
Received 23 May 2023; Received in revised form 12 July 2023; Accepted 30 July 2023
Available online 10 August 2023

on the total length and on the number of changes of direction (*elbows*) in the path followed by the pipes and cable trays. Although initially these paths must be traced in a continuous three-dimensional region, a discretization of the entire space is desirable in order to mathematically address the problem. One of the possible options is to discretize the space by means of a graph structure. Thus, in case no further additional constructability constraints are required and a single pipeline or cable tray is to be routed, this problem is equivalent to finding an optimal path between two nodes in a graph, and solutions can be found in polynomial time using Dijkstra's algorithm (Dijkstra, 1959). However, in these types of problems many pipelines and cable trays must be routed together, and specific constructability constraints are also required. For instance, the length of the cable that can be accommodated when laying lines (wiring tray) should be upper bounded, a minimum distance between consecutive elbows of the same cable tray, and between different cable trays and pipelines, must be assured, obstacles must be avoided, some zones are preferable, etc., making the problem extremely hard to solve. Therefore, the PCTLP poses a significant mathematical and computational challenge.

In this paper, we propose a suitable mathematical optimization-based tool for the PCTLP in naval design, considering not only the joint arrangement of several elements of the same type through cable trays but also the technical feasibility requirements of the design.

1.2. Literature review

The pipeline location problem in naval design has been widely considered in the literature since it is a fundamental and challenging problem in the design of large ships. The main contribution of most proposals for this problem is the development of metaheuristic algorithms for determining the *optimal* routes of multiple pipelines in an intricate and obstructed three-dimensional (3D) space. There are two elements that are shared among the different works on this topic. Firstly, most of the works agree that the most efficient way to handle this problem in a continuous region (the 3D ship layout) is by appropriately discretizing the solution space. Thus, the first phase is to derive a suitable discrete space to represent the feasible solutions for the problem. Secondly, most of the approaches are based on decomposing the multiple pipeline routing problem into single pipeline problems and applying a shortest-path-like algorithm to solve them.

Concerning the discretization of the space, different strategies have been proposed. In Lee (1961), the author develops a method based on partitioning the region into square cells and removing cells containing obstacles. Then, paths are constructed in the remaining continuous regions. This technique has been applied to the pipeline routing problem for several authors (see, e.g., Ando and Kimura, 2011; Asmara, 2013; Asmara and Nienhuis, 2006; Kim et al., 2013). On this domain, one can apply several strategies to construct pipeline paths, as a maze algorithm (Lee, 1961; Rourke, 1975), or an escape algorithm (Hightower, 1969). On the other hand, Guirardello and Swaney (2005) propose a network-based discretization scheme to reduce the dimension of the solution space, where each vertex in the network models a junction of a feasible piperack structure. In this space, routing a single pipeline reduces to finding the shortest path between its source and its target. In Park and Storch (2002), the authors propose a mixture of the above two methods, where a network-based cell-generation method is developed.

Jiang et al. (2015) propose an ant colony optimization heuristic for the pipeline routing problem. In Dong et al. (2022), a multi-objective framework for the pipe route design is considered. More recently, Blanco et al. (2022) address the problem of routing multiple pipelines using an adapted min-cost flow formulation and propose different matheuristic algorithms.

In most of the previous works, the pipelines are routed while avoiding overlapping. However, in realistic situations, it is allowed to jointly route similar elements through cable trays. These devices help reduce the total cost and required space. As a result, they have been recognized as a common and advisable practice, not only in naval design (Park et al., 2020) but also in building constructions (Pogorelskiy and Kocsis, 2022). Nevertheless, the literature on the PCTLP is scarce. Once the clusters of cables/pipelines to be routed through the same cable tray are determined, one can apply any of the available approaches for the pipeline routing problem. Castorani et al. (2018) propose a two-phase approach for this problem. In the first phase, the arrangement of the cables within the trays is decided, and in the second phase, the routing of the cable trays is determined using a genetic algorithm. This strategy may provide suboptimal solutions since the determination of the routes and the decisions on the contents of the trays are computed separately.

In this paper, we analyze the PCTLP in a single mathematical optimization model for the first time and provide efficient strategies to obtain high-quality solutions. This model extends the one proposed in Blanco et al. (2022) by incorporating cable trays. The inclusion of cable trays in the design of pipeline routes significantly increases the complexity of the problem. This complexity arises from the need to determine not only the routes of the pipelines but also the placement of the trays and the allocation of pipelines or cables to each tray, considering the technical requirements of both the pipelines and the trays. All these decisions are optimally made by minimizing the overall space occupied by the pipelines and the cable trays.

1.3. Contributions

The main contributions of this work are:

- A mathematical programming formulation for the PCTLP as a generalized Minimum Cost Multicommodity Network Flow Problem incorporating several technical requirements.
- The design of an exact *ad hoc* relax-and-cut procedure for solving the problem by relaxing some of the difficult technical constraints.
- A novel family of math-heuristic approaches for the problem based on two phases: search of potential pipeline and cable trays routes and transformation of these initial routes into feasible pipeline and cable trays routes verifying the technical requirements. Different strategies are proposed for each of the phases that can be adequately combined to derive different heuristic methods suitable to obtain good quality feasible solutions in reasonable CPU time.
- Extensive computational experiments in synthetic instances in order to compare the different solution approaches.
- The validation of our proposals on a real instance provided by our industrial partner.

1.4. Organization

The paper is organized as follows: Section 2 describes the main elements involved in the PCTLP and their mathematical representation. Section 3 presents the mathematical programming model for the PCTLP in naval design, as well as the relax-and-cut method proposed for solving the problem optimally. Several matheuristic algorithms are provided in Section 4, which can be combined to solve large-sized instances. In Section 5, we report the results of computational experiments. Additionally, a case study built from a real scenario provided by our industrial partner is included. Finally, in Section 6, we present the conclusions drawn from this work and outline future research directions.

2. Main elements of the PCTLP

In the PCTLP, the objective is to route multiple pipelines and cable trays (services) optimally through a complex and obstructed space (solution space), while minimizing certain design costs. In this section, we will describe the key elements of the PCTLP and establish the notation that will be used throughout the rest of the document.

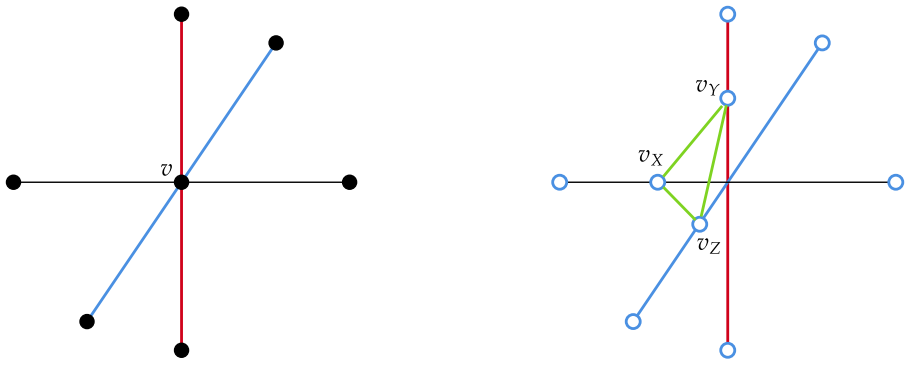

Fig. 1. A depiction of nodes exploited and virtual edges.

2.1. Services

In naval design, PCTLP involves connecting several devices or equipment by means of pipelines and cables giving rise to a finite set of services of different nature (gas pipes, water pipes, drainage pipes, optical fiber cables, power cables, air conditioning pipes, etc.) that are to be located. Each of these services is classified into a type from a finite set $\mathcal{T}$, so that, for a given $t \in \mathcal{T}$, services of type t are allowed to be jointly routed through the same tray. On the other hand, services of different types must ensure certain distance-based requirements. For instance, services such as power, phone, or fiber optic cables are considered to belong to the same type t and it is reasonable to assume that they can be traced jointly in the same tray. In contrast, services such as water pipes and power cables clearly belong to different types of services and must be conveniently separated in the design.

We assume that each service of type $t \in \mathcal{T}$ is characterized by a finite set of sources, each of them with a given integer amount of out-flow, and a finite set of sinks, each of them with a given integer amount of in-flow, such that the overall out-flow coincides with the overall in-flow. For each type $t \in \mathcal{T}$, this flow represents the number of services of type t to be routed.

We will denote by $\{s_1^t, \dots, s_{p_t}^t\}$ the set of sources, by $\{d_1^t, \dots, d_{q_t}^t\}$ the set of sinks, by $g_{s_i^t}$ for $i = 1, \dots, p_t$, the number of services of type t sent from origin s_i^t (out-flow at source s_i^t) and by $g_{d_j^t}$, for $j = 1, \dots, q_t$, the number of services of type t sent to destination d_j^t (in-flow at sink d_j^t) assuming that, $\sum_{i=1}^{p_t} g_{s_i^t} = \sum_{j=1}^{q_t} g_{d_j^t}$ $\forall t \in \mathcal{T}$.

2.2. Solution space

Designing the routing of pipelines and cable trays in a naval structure requires us to consider as solution space a 3D region that is obtained by removing a set of polyhedral obstacles from a cuboid (representing a cabin in the ship). As usual in the literature (see, e.g., Ando and Kimura, 2011; Asmara, 2013; Lee, 1961), we discretize that continuous space by creating a 3D grid containing the set of sources $\{s_1^t, \dots, s_{p_t}^t\}$ and the set of destinations $\{d_1^t, \dots, d_{q_t}^t\}$ for each type $t \in \mathcal{T}$. They are nodes of the grid, together with other nodes and links connecting them. The pipeline and cable tray routes will be created by constructing in the grid paths linking sources and destinations of the services of each type $t \in \mathcal{T}$. To take into account costs derived from direction changes (elbows) happening in these paths, we modify the original grid by transforming each physical node, v, into three virtual nodes (v_X, v_Y, v_Z), with the same 3D coordinates, representing the three possible directions to take. These virtual nodes associated to the same physical node are linked between them through new links, that we call *virtual links*, with lengths equal to zero. The links in the original grid keep their length but now they only connect virtual nodes representing the same direction (links parallel to the X-axis connect v_X nodes, links parallel to the Y-axis connect v_Y nodes and links parallel to the Z-axis connect v_Z nodes).

Fig. 1 shows (on the left) a central node with six other nodes connected to it, and three exploited nodes that originate from the central node (on the right). Each of the virtual nodes is only connected to two neighboring nodes sharing edges parallel to a particular axis and two other virtual nodes that correspond to the same physical node. Fig. 2 demonstrates how the graph can simulate different types of turns along a path. The left image shows a path that requires no elbows and only involves one exploited node, while the center and right images require traversing virtual edges, resulting in additional costs for using elbows.

Let $\widetilde{G} = (V, E)$ be the undirected graph obtained after modifying the original grid described above, where V represents the set of virtual nodes and E represents the set of edges linking them. As already mentioned, the set E includes the virtual edges used to model elbows, denoted by E^v, as well as the replicas of the physical edges that now connect adjacent virtual nodes in the same direction.

Given a virtual edge $e \in E^v$, we denote by $RE(e) = \{\widetilde{e} \in E^v : \widetilde{e}$ has the same coordinates as $e\}$, namely the set of edges linking virtual nodes associated to the same physical node as the one linked by e. We also denote by $d_{ee'}$ the minimum Euclidean distance between the edges $e, e' \in E$ and by $G = (V, A)$ the directed version of the graph $\widetilde{G}$, i.e., $A = \{(i, j) \cup (j, i) : \ e = \{i, j\} \in E\}$ is the arc set induced by E.

2.3. Cost structure

Different costs are considered in the design of the routes of the pipelines and trays (as the length of the pipelines/trays, the number of elbows, the height of the pipelines in the cabin, etc.). These costs are incorporated to the graph structure described above in different layers, for each of the types to be routed.

For each $e \in E$ and each $t \in \mathcal{T}$, we denote by c_e^t the cost of using edge e to route a service of type t. If $e \in E^v$ is a virtual edge, this cost includes a positive cost for elbow use but not a length cost, i.e., routes using virtual edges only imply that a change of direction occurred and an extra cost for elbow use is incurred. On the contrary, for the replicas of the physical edges, $e \in E \setminus E^v$, this cost includes a positive length cost, but not an elbow utilization cost.

2.4. The pipelines and cable trays location problem

The PCTLP in a ship that we analyze here, consists of routing the different types of services, by determining the location of the possible trays concentrating the flow of the services of the same type in part of the route, and the different connections between the pipelines/cables and the trays by minimizing the overall design cost, as well as verifying the technical constructability requirement. Specifically, a minimum distance between consecutive elbows of a pipeline/tray and a minimum security distance between services of different types are required.

We denote by D^t the minimum allowed distance between consecutive elbows on a pipeline/cable tray of services of type $t \in \mathcal{T}$, by

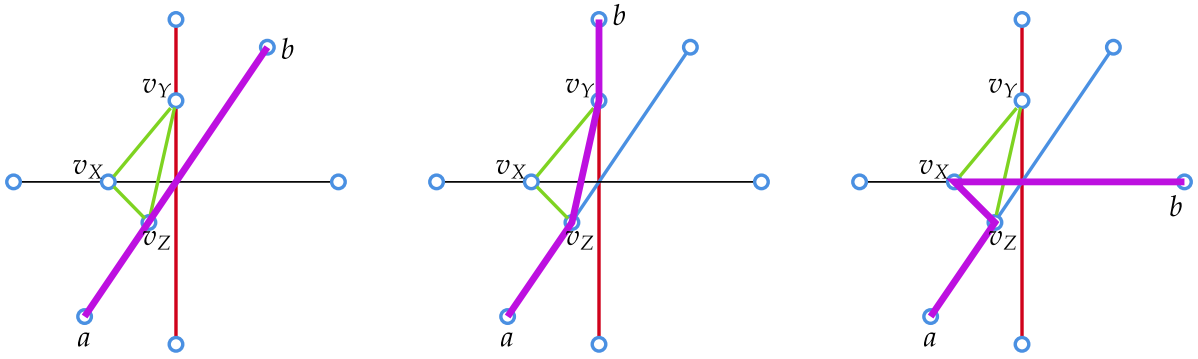

Fig. 2. A visual representation showcasing how nodes exploited and virtual edges can be employed to simulate various forms of turns in a path connecting node a and node b.

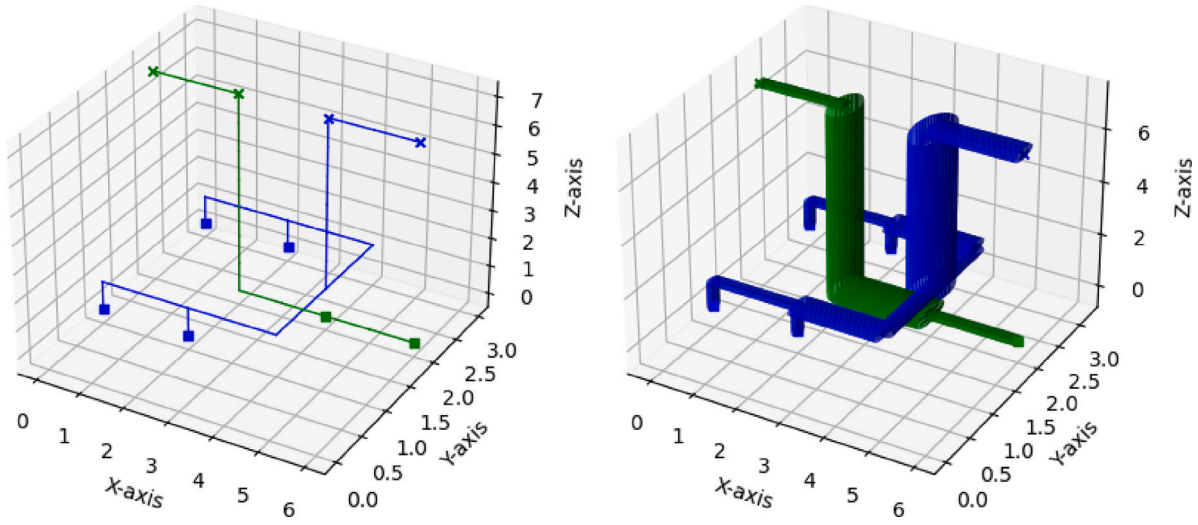

Fig. 3. Feasible solution for a small instance of the PCTLP: as a union of edges in the graph (left) and its 3D actual representation (right).

R^t the width of the pipeline/cable of a single service of type t, by Δ^t the minimum distance between services of type t and services of any other type, and by $\Delta^{tt'} := \max\{\Delta^t, \Delta^{t'}\}$ the minimum distance between services of type t and t'.

In Fig. 3, we show a feasible PCTLP solution for a small example. There, we assume that services of two types are to be routed (services of color `blue` and `green`). Services of type `blue` have four different origins (each of them with out-flow 1) and two sinks (each of them with in-flow 2) available to route flows, while services of type `green` have two origins and two sinks (two sources with out-flow of 2 and 1 each, and two sinks with in-flow 1 and 2 separately). In the left picture, the origins and sinks are depicted as squares (□) and crosses (×), respectively. As can be observed in the picture, the services of the same type are jointly routed in a tray in part of their paths from the origins to the sinks. This solution is drawn (left) as a union of edges (a path) in the graph, but they represent dimensional pipelines or trays in the cabin of the ship. This might cause incompatibilities between the different types represented in each edge if the minimum safety distance between services of different types is not met. In Fig. 3 (right), we show the 3D representation of the solution once we have adapted the volume of the pipelines/trays to their actual size.

3. A mathematical programming model for the PCTLP

In this section, we describe the mathematical programming model that we propose for the PCTLP described in the previous section. We model the problem as an adapted **M**inimum **C**ost **M**ulticommodity **N**etwork **F**low **P**roblem (MCMNFP) with additional constraints, that include, among other features, the pipelines/cable trays layout. The goal of MCMNFP is to simultaneously route all the types of services from sources to sinks, through a network at minimum cost. This model has been widely studied in the literature and has been applied to different industrial problems (Garg and Smith, 2008; Guimarães and Prata, 2022). The interested reader is referred to Salimifard and Bigharaz (2022) and the references therein for further details on the MCMNFP.

In our model, (integer) flow variables, f_{ij}^t are defined for each type $t \in \mathcal{T}$ and each arc $(i,j) \in A$. We assume that a single unit of flow is to be routed for each service of type $t \in \mathcal{T}$ and thus, these variables indicate the number of services of type $t \in \mathcal{T}$ routed through the arc $(i,j) \in A$.

We denote by F^t the maximum flow of type $t \in \mathcal{T}$ allowed to be routed along any edge of the graph, i.e., the maximum number of services of type $t \in \mathcal{T}$ that can be jointly routed in the same cable tray.

Flow variables together with the following binary variables, also used in our model, allow us to detect the routes of each of the pipelines/cable trays in the solution.

$$x_e^t = \begin{cases} 1 \text{ if edge } e \text{ is used by the route of services of type } t, \\ 0 \text{ otherwise.} \end{cases} \qquad \forall e \in E,\ \forall t \in \mathcal{T}.$$

Let $V' \subset V$ be the subset of vertices in the graph which are neither sources nor sinks and let $D^{tt'} > 2R^tF^t + 2R^{t'}F^{t'} + \Delta^{tt'}$ be an upper bound on the width occupied by two pipelines/cable trays, one of them of type t and the other one of type t'.

Table 1 summarizes the main elements and variables used in our model. Once these elements are set up, we can formulate the PCTLP as

Table 1
Parameters and variables used for formulating PCTLP.

Parameters	
V	set of virtual nodes.
E	set of total edges.
E^v	set of virtual edges.
$RE(e)$	$\{\widetilde{e} \in E^v : \widetilde{e}$ has the same coordinates as $e \in E^v\}$.
A	arc set induced by E.
$\mathcal{T}$	set of possibles service types.
$\{s_1^t, \dots, s_{p_t}^t\}$	set of sources for services of type $t \in \mathcal{T}$
$\{d_1^t, \dots, d_{q_t}^t\}$	set of sinks for services of type $t \in \mathcal{T}$
$V' \subset V$	subset of vertices which are neither sources nor sinks.
$g_{s_i^t}$	number of services of type t sent from source s_i^t (out-flow at s_i^t).
$g_{d_j^t}$	number of services of type t sent to sink d_i^t (in-flow at d_j^t) .
c_e^t	cost of using edge e by a service of type t.
R^t	width of the pipeline/cable of a single service of type t.
$D^{tt'}$	upper bound on the width occupied by two pipelines of type t and t'.
D^t	minimum allowed distance between consecutive elbows on a pipeline of type t.
Δ^t	minimum allowed distance between services of type t and services of any other type.
$\Delta^{tt'}$	minimum allowed distance between services of type t and t'.
$d_{ee'}$	minimum Euclidean distance between the edges $e, e' \in E$
F^t	maximum allowed number of services of type t jointly routed in the same cable tray.
Variables	
x_e^t	$\begin{cases} 1 & \text{if edge } e \text{ is used by the route of services of type } t, \\ 0 & \text{otherwise.} \end{cases}$
f_{ij}^t	amount of flow of pipeline of type t routed through arc (i, j).

the following **M**ixed **I**nteger **L**inear **P**rogramming (MILP):

$$\min \sum_{e \in E} \sum_{t \in \mathcal{T}} c_e^t x_e^t \tag{1}$$

$$\text{s.t.} \sum_{\substack{j \in V': \\ (i,j) \in A}} f_{ij}^t - \sum_{\substack{j \in V': \\ (i,j) \in A}} f_{ji}^t = 0, \quad \forall t \in \mathcal{T}, \forall i \in V', \tag{2}$$

$$\sum_{\substack{j \in V: \\ (s_i^t, j) \in A}} f_{s_i^t j}^t - \sum_{\substack{j \in V: \\ (j, s_i^t) \in A}} f_{j s_i^t}^t = g_{s_i^t}, \quad \forall t \in \mathcal{T}, i = 1, \dots, p_t, \tag{3}$$

$$\sum_{\substack{i \in V: \\ (i, d_j^t) \in A}} f_{i d_j^t}^t - \sum_{\substack{i \in V: \\ (d_j^t, i) \in A}} f_{d_j^t i}^t = g_{d_j^t}, \quad \forall t \in \mathcal{T}, j = \cdots, q_t, \tag{4}$$

$$(f_{ij}^t + f_{ji}^t) R^t + (f_{i'j'}^{t'} + f_{j'i'}^{t'}) R^{t'} + \Delta^{tt'} \le d_{ee'} + D^{tt'}(2 - x_e^t - x_{e'}^{t'}),$$
$$\forall e = \{i, j\}, e' = \{i', j'\} \in E, \forall t, t' \in \mathcal{T}, \tag{5}$$

$$\sum_{\tilde{e} \in RE(e)} x_{\tilde{e}}^t + \sum_{\tilde{e} \in RE(e')} x_{\tilde{e}}^t \le 1,$$
$$\forall e, e' \in E^v : 0 < d_{ee'} \le D^t, e \ne e', \forall t \in \mathcal{T}, \tag{6}$$

$$x_e^t \le f_{ij}^t + f_{ji}^t \le F^t x_e^t, \quad \forall t \in \mathcal{T}, \forall e = \{i, j\} \in E, \tag{7}$$

$$x_e^t \in \{0, 1\}, \quad \forall e \in E, \forall t \in \mathcal{T}. \tag{8}$$

Objective function (1) minimizes the overall cost of the routes. Note that this objective function benefits the creation of cable trays since the cost of using an edge does not depend on the number of services using this edge (edge flow) but only if it is used or not. Constraints (2), (3), and (4) are the flow conservation constraints ensuring the adequate construction of paths for the services. Constraints (5) ensure that in a feasible solution to the problem, the minimum allowed distances between services of a different type are kept considering the pipelines/cable tray width, which depends on the flow. The expression $R^t(f_{ij}^t + f_{ji}^t) + R^{t'}(f_{i'j'}^{t'} + f_{j'i'}^{t'}) + \Delta^{tt'}$ for a pair of edges $e = \{i, j\}$ and $e' = \{i', j'\}$ and types $t, t' \in \mathcal{T}$, models the width of the space occupied by the two pipelines/cable trays (plus the safety distance $\Delta^{tt'}$). If both e and e' are used to route services of type t and t', respectively, this amount cannot exceed the minimum Euclidean distance between the edges, since otherwise, the edges are not far enough to fit the two pipelines/cable trays.

Constraints (6) assure that in case two elbows of the same pipelines/cable trays are closer than the minimum allowed distance (D^t), only one of them can be activated in the solution (*elbow test*). Clearly, given two edges $e, e' \in E^v$, if $d_{ee'} \le D^t$, all the edges in $RE(e)$ and $RE(e')$ are incompatible and then cannot take simultaneously a value of 1. Finally, constraints (7) (together with constraints (5)) impose that, if edge $e = \{i, j\}$ is used by the route of a service of a given type $t \in \mathcal{T}$ then, only services of type $t \in \mathcal{T}$ can be routed through the arcs (i, j) and (j, i). They also force to activate edge $e = \{i, j\}$ for routing services of type t if there exists a service of type t routed through the edge and limit the number of services of type t routed through the edge to be at most F^t.

The solution of the PCTLP is a set of paths connecting sources with destinations where services of different type do not overlap on the graph G (nor in the 3D representation of the pipelines/cable trays). Ideally, in this solution, several services of the same type should join and go together through various edges to eventually separate and finish on their sinks.

3.1. An exact relax-and-cut procedure for solving PCTLP

Note that the number of constraints in families (5) and (6) is too large to make it possible to include them all into a MILP solver for medium/large-sized instances. To overcome this issue, we start by solving a relaxed formulation without these constraints and, to avoid the infeasibility of the solutions, we proceed by incorporating them as needed in a relax-and-cut procedure. More precisely, the solution of the relaxed problem may not meet the requirements of constraints (5) and/or (6). To make sure each solution is valid, we check for feasibility by using an enumerative procedure. If a constraint is found to be violated, it is separated and added to the constraint pool. Then, we solve again the relaxed formulation of the problem obtained after adding the constraints in the pool. The procedure terminates when a solution is found verifying all the constraints.

To check if constraints (5) are met, we measure the minimum distance between edges belonging to different cable trays. If this distance is smaller than the sum of the width of the involved cable trays and the required security distance, we consider the constraint to be violated and add it to the constraint pool.

To verify the validity of constraints (6), we first arrange the virtual edges in the path of a pipeline/cable tray and then measure the minimum distance between consecutive virtual edges. If this distance is smaller than or equal to the minimum required distance between elbows for the pipelines/cable tray type, the constraint is considered

violated and it is introduced as a feasibility cut to the set of constraints pool.

Note that the size of the *activated* edges (by x) and flows (f) in a relaxed solution of the problem for a given type t is much smaller than the number of edges and arcs in the original graph, and thus, the separation process can be efficiently solved in practice. Furthermore, we embed this procedure, into a branch-and-cut scheme using callbacks, which are available in most of the commercial off-the-shelf software, such as Gurobi, CPLEX, or FICO.

4. A general heuristic approach

Although the relax-and-cut procedure mentioned in the previous section alleviates the problem solving caused by the difficulty of the technical constraints, the number of variables and constraints of the model is still huge even for medium-sized instances. In this section, we develop a family of heuristic approaches designed to provide feasible solutions in lower computation times than the exact relax-and-cut procedure.

The different heuristic approaches that we propose are based on the following two main phases:

Phase I: Construction of initial paths and cable trays. In this first step, the goal is to construct promising paths in the graph not necessarily feasible connecting the sources and sinks of the services of each type and encouraging services of the same type to share edges using the overall less costly edges.

Phase II: Transformation to feasible paths and cable trays. Given a partial solution obtained in Phase I, the aim of this stage is to check for feasibility (and to provide feasible solutions if necessary) of the PCTLP by correcting the technical requirements, i.e., the minimum distance between services and the elbow test.

In what follows, we describe the different strategies that we propose to deal with the different phases above.

4.1. Strategies for phase I

For this phase, we propose two different approaches to enforcing or encouraging the construction of cable trays.

In the first strategy, the so-called *Constructive Algorithm* (H1) locates certain points (called *supernodes*) that will define the beginning- and end-nodes of the pipelines/cable trays. Then, both supernodes of each cable tray are connected by a path, constructed by an adapted shortest path algorithm. Finally, connections (not necessarily verifying the technical requirements) between sources, sinks, and trays are established.

The second strategy, the so-called *Decomposition Algorithm* (H2), is based on solving separately for each type of service the **M**inimum **C**ost **M**ulticommodity **N**etwork **F**low **P**roblems (MCMNFP) which is solvable in polynomial time by a primal network simplex algorithm (see, e.g., Orlin, 1997). The different types of services are independently routed, and the construction of cable trays is encouraged by increasing the costs of certain edges in the underlying graph.

The two approaches are independently performed for each type $t \in \mathcal{T}$ and are detailed as follows:

Constructive Algorithm (H1).

For each type $t \in \mathcal{T}$:

- **H1.1 (Supernodes location).** In order to find the supernodes (among the set of sinks and sources) which are the end-nodes of the q cable trays that are constructed, we solve a mathematical programming model for each service type based on a p-median location problem. We use the following set of binary variables:

$$y_j^l = \begin{cases} 1 & \text{if node } j \text{ is selected as the } l\text{th supernode,} \\ 0 & \text{otherwise,} \end{cases}$$

where $l \in L = \{1, \dots, 2q\}$, $q = \lceil \phi^t / F^t \rceil$ and ϕ^t is the total flow that type t services transport through them (number of services of type t). Also, we consider the following set of binary variables:

$$z_{ij}^l = \begin{cases} 1 & \text{if source/sink } i \text{ is assigned to} \\ & \text{the } l\text{th supernode located at node } j, \\ 0 & \text{otherwise.} \end{cases}$$

Let $ST = \{s_1^t, \dots, s_{p_t}^t\} \bigcup \{d_1^t, \dots, d_{q_t}^t\}$ be the total set of sources and sinks for service type t. The model is formulated as:

$$\min \sum_{l \in L} \sum_{j \in V} \sum_{i \in ST} |h_i| d_{ij} z_{ij}^l \tag{9}$$

$$\text{s.t.} \sum_{l \in L} \sum_{j \in V} z_{ij}^l = 1, \quad \forall i \in ST, \tag{10}$$

$$z_{ij}^l \le y_j^l, \quad \forall l \in L, \forall i \in ST, \forall j \in V, \tag{11}$$

$$\sum_{l \in L} y_j^l \le 1, \quad \forall j \in V, \tag{12}$$

$$\sum_{j \in V} y_j^l = 1, \quad \forall l \in L, \tag{13}$$

$$\sum_{l \in L} \sum_{j \in V} y_j^l = 2q, \tag{14}$$

$$\sum_{l \in L} \sum_{i \in ST} h_i z_{ij}^l \le F^t y_j^l, \quad \forall j \in V, \tag{15}$$

$$\sum_{i \in ST} \sum_{j \in V} h_i z_{ij}^l = -\sum_{i \in ST} \sum_{j \in V} h_i z_{ij}^{l+1}, \quad \forall l \in L : l \text{ is odd}, \tag{16}$$

$$y_j^l \in \{0,1\}, \quad \forall l \in L, \forall j \in V, \tag{17}$$

$$z_{ij}^l \in \{0,1\}, \quad \forall l \in L, \forall i \in ST, \forall j \in V. \tag{18}$$

We show in Table 2 a summary of the parameters and variables used in our formulation. The objective function (9) minimizes the flow per path length between sources/sinks and supernodes, where h_i is the flow of the source/sink i and d_{ij} is the shortest path distance from source/sink i to supernode j. Constraints (10) force each source/sink to be assigned to one supernode. Constraints (11) ensure sources/sinks can only be assigned to an activated supernode. Constraints (12) guarantee there cannot be more than one supernode with the same label. Constraints (13) assign each label $l \in L$ to a single supernode. Constraint (14) ensures there is a total of $2q$ supernodes. Constraints (15) guarantee that the total flow that reaches a supernode must be less than its capacity F^t. Constraints (16) match supernodes so they have the same flow of opposite sign. If F^t is large enough, then only one tray of type t is formed. In this case, $L = \{1\}$, $q = 1$ and constraints (11), (12), (14), (15) are ignored.
- **H1.2 (Supernodes connection).** To connect supernodes of each service type through feasible cable trays we use an adapted shortest path-based method similar to the one proposed in Blanco et al. (2022). This approach ensures that the pipelines/cable trays verify all the technical requirement imposed in our problem (elbow-test and minimum distance between different types).
- **H1.3 (Linking sources/sinks to trays).** Every source/sink is connected to its corresponding tray using the shortest path, which is also considered as part of the cable tray and can be used to connect other sources/sinks to it. This process is repeated until all the sources/sinks are connected to

Table 2
Parameters and variables of formulation (9)-(18).

Parameters	
h_i	flow of the source/sink i.
d_{ij}	minimum distance between edges i and j.
ϕ^t	number of services of type t.
F^t	maximum number of services of type $t \in \mathcal{T}$ that can be jointly routed in the same cable tray.
q	$\lceil \phi^t / F^t \rceil$.
L	$\{1, \dots, 2q\}$.
V	vertices in the graph.
ST	total set of sources and sinks for service type t.
Variables	
z^l_{ij}	$\begin{cases} 1 & \text{if source/sink } i \text{ is assigned to the } l\text{th supernode located at node } j, \\ 0 & \text{otherwise.} \end{cases}$
y^l_j	$\begin{cases} 1 & \text{if node } j \text{ is selected as the } l\text{th supernode,} \\ 0 & \text{otherwise.} \end{cases}$

a tray. We denote by E_t^{best} the list of edges obtained at the end of the procedure. Note that these connections may not verify the technical requirement imposed in the PCTLP.

Decomposition-based algorithm (H2).

For each type $t \in \mathcal{T}$:

- **H2.1.** A MCMNFP is solved only for type t. We denote by E_t the list of edges induced by the solution (arcs with a positive flow). Observe that since the remainder types are not considered in the problem, the technical constraints are not guaranteed.
- **H2.2.** For each connected component of E_t, we increase the cost of all the edges that do not belong to the selected connected component and we solve again the MCMNFP for this type of service t. This boosts all the services to use the same set of edges (those in the selected connected component). Once it is repeated for each connected component, the best solution (based on the objective value of the PCTLP), E_t^{best}, is kept. This strategy allows to concentrate the flow in the overall less costly set of edges.

4.2. Strategies for phase II

For this phase, we propose two different iterative approaches in order to construct feasible PCTLP solutions, verifying all the technical requirements. Both approaches use as starting solution the one obtained in Phase I, i.e., E_t^{best}. The first algorithm, the so-called *Graph trimming*, is based on solving the PCTLP on a trimmed graph of the original one, and so with a smaller number of variables and constraints. The second approach, the so-called *Increasing Cost*, consists of increasing the costs of the edges not verifying the technical constraints to avoid their use, and repeating the process until they are fulfilled.

Graph Trimming (GT). Once the initial heuristic solution is obtained from any of the strategies described in Phase I, constraints (5) and (6) are evaluated. If they are verified, the initial heuristic solution is feasible for the PCTLP. Otherwise, we propose an iterative procedure that begins with the initial heuristic solution. Then, for each service type, a cylindrical region of a given initial width is considered around each edge of the union of edges that take part of the initial solution. Instead of solving the entire PCTLP in the graph, we solve it in a reduced space, the union of these regions by fixing the variables indicating arcs outside of these regions to zero. If the problem is feasible, it provides a feasible solution, otherwise, the width of the regions is increased and the process is repeated until feasibility is achieved. Note that as the number of iterations increases the dimension of the trimmed graph becomes larger, and then, the problem requires more computational time to be solved. Although in the worst-case scenario, the algorithm requires solving the original instance of the problem (the whole graph), in practice, this strategy significantly reduces the number of variables needed to solve the problem.

Increasing Cost (IC). This strategy consists of guiding the solutions obtained by the MCMNFP to verify all the requirements in the PCTLP by increasing the cost of the non-desired edges in the design (to the maximum cost value). The algorithm considers a maximum number of iterations to be performed, *MaxIt*, and for each iteration $it \in \{1, \dots, MaxIt\}$ a sorted list of all the commodities (types), $T_{\text{sort}}(it)$. A multistart approach is then applied by testing different sorted lists and choosing the best of them. Some of the sorted lists are based on the decision maker's preferences and others are obtained randomly. This approach, for iteration $it \in \{1, \dots, MaxIt\}$ and for type $t \in T_{\text{sort}}(it)$, consists of the following iterative procedures:

- **Elbow-Test.** The elbow constraints (6) are checked for the obtained solution. If any of them is violated one of the two elbows not verifying the test is chosen and its cost is increased to avoid its use in later elbow-test iterations. The MCMNFP is again solved with the new costs and E_t^{best} is updated accordingly. This step is repeated until the elbow test is verified or a maximum number of elbow-test iterations is reached (in whose case the solution in iteration it is not valid).
- **Minimum Distance.** The distance requirements (5) are tested by checking overlapping edges with respect to each of the previously processed types $t' \in \{1, \dots, t-1\}$, if any. If edge e used in type t with flow f_e^t overlaps edge e' used in any type $t' \in \{1, \dots, t-1\}$ with flow $f_{e'}^{t'}$, this stage is addressed by increasing, for type t, the cost of the edges in conflict, that is, the edges at a distance lower than or equal to $f_e^t R^t + f_{e'}^{t'} R^{t'} + \Delta^{tt'}$ from the edges used in $E_{t'}^{best}$ and then, go to Phase I to construct a new list of edges for type t, E_t^{best} (updated). This procedure is repeated until there are no conflicts or a maximum number of overlapping iterations (MaxOverIt) is reached (in whose case the solution, in iteration it, is not valid). Once no conflicts appear, to prevent overlapping for the next types in the sorted list, the cost for types $t+1, \dots, T$ is increased for the edges in E_t^{best}.

Once all iterations are solved we keep the solution of the best iteration (based on the objective value of the PCTLP).

Note that the strategies described above for the two different phases of the heuristic can be combined in different ways. In Fig. 4, we illustrate

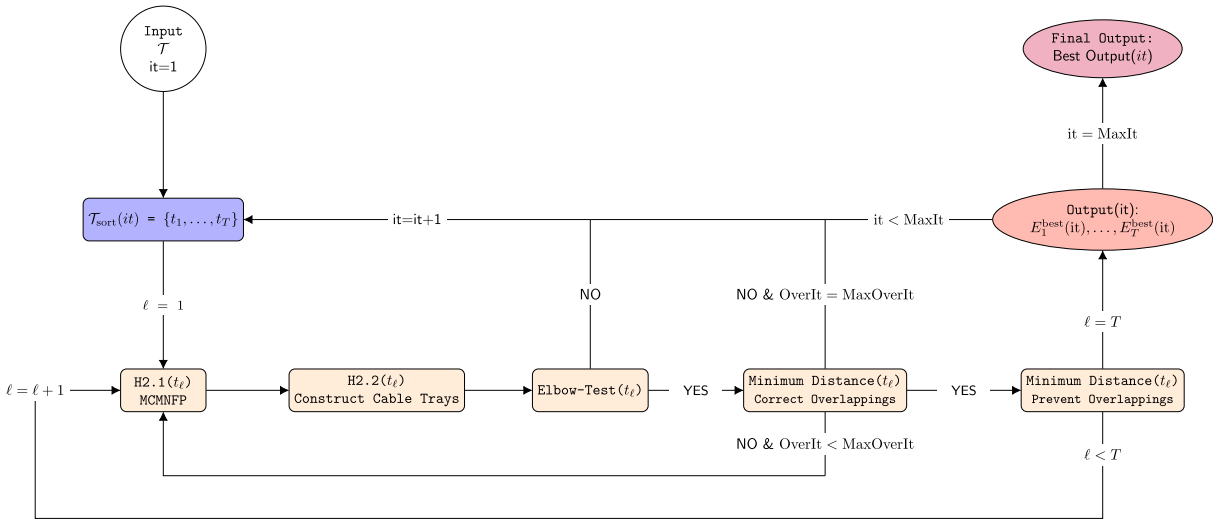

Fig. 4. Flowchart of the heuristic combination of H2 for Phase I and Increasing Cost for Phase II.

a possible choice in which H2 is applied for Phase I and Increasing Cost for Phase II. Note that since in Phase II, the types of services are sorted, the sequence in which the types are processed in Phase I follows the same order.

5. Computational experience

In this section, we report the results of our computational study aimed at evaluating the effectiveness of our proposed methods for solving the PCTLP. Our computational experience is organized into three parts. First, we examine the computational performance of our algorithms on a battery of randomly generated instances. Second, we assess the performance of the algorithms on instances that we have specifically designed for the PCTLP, the so-called "corridor instances". These instances are publicly available at https://github.com/anticiclon/Dataset-Trays-Location-Problem. Finally, we apply our methodology to a real-world case study provided by our industrial partner.

We use Gurobi v7.7 optimizer to solve all instances on a Windows 10 platform, powered by an Intel(R) Core(TM)i7 CPU 2.93 GHz processor and 16 GB RAM. We use the default settings for all Gurobi solver parameters and set a CPU time limit of 4 h.

5.1. Random instances

In our first study, we consider a set of random instances, that have been already used in the literature, to validate our proposal. Each instance consists of a cube of length 128 units containing a number of obstacles and different services that must be routed between a set of sources and sinks (see Table 3). More specifically, for each instance an orthogonal 3D grid is built on such a cube with a density $d \in \{17, 33\}$, with fixed distances between nodes, i.e., each axis is divided into 16 or 32 equally sized parts, and the grid is constructed. Five groups of $o \in \{5, 10, 15\}$ 10-units cubic obstacles are randomly generated. Five groups of $s \in \{5, 8, 12\}$ services are also generated each of them classified into one of two types of service (A or B), such that type A will have services enumerated with an odd number whereas B has those services enumerated with an even number. The generated grid is constructed such that it has enough capacity to gather all services of the same type. In scenarios with 5 services, the width of a service in both types is $R^t = 0.33$. Analogously, in scenarios with 8 or 12 services widths are $R^t = 0.25$ and $R^t = 0.17$, respectively.

We take into consideration several factors when calculating the cost function, which include physical distance, the number of bends or elbows in the path, and changes in height. More precisely, we define the cost of an edge e belonging to type t as follows:

$$c_e^t = \alpha_1^t (d_e + 10El_e + 2Ch_e),$$

where α_1^t is a random integer ranging in $[1, 9]$, d_e is the length of edge e, El_e is equal to 1 if edge e has an elbow, and 0 otherwise, and Ch_e is equal to 1 if edge e is a vertical edge along the Z-axis, and 0 otherwise.

In total, 90 benchmark instances are generated based on the combination of distances, services, and obstacles. Table 3 summarizes the types of instances used in this study.

Table 3
Summary of the types of random instances.

Density (d)	#Services (s)/Radii (R)	Obstacles (o)
17, 33	5/0.33 8/0.25 12/0.17	5, 10, 15

The size of each instance can be described by the number of edges in the resulting graph. For a given density d, this graph has $3(d-1)d^2$ physical and $3d^3$ virtual edges.

Finally, we recall that for each case, the obstacles generated within the grid remove some edges. Since there is one binary variable and two flow variables for each edge and each type, the final average number of edges can also be obtained by dividing column `#Vars` by six.

The proposed general heuristic approach described in Section 4 is very flexible and can be parameterized in many different ways. Among all possible combinations of strategies for Phases I and II, we have run two of them and, additionally, a mixing strategy to compare their performance against the exact relax-and-cut (EX) namely:

H2+IC: We test the decomposition-based algorithm (H2) improving the results with the increasing cost method in order to guarantee the elbow-test and minimum distances between services.

H12+GT: We trim the graph around the solutions provided by H1 and H2 and solve the PCTLP in the trimmed graph with a time limit of 900 s. This time limit is extended if required until a feasible solution is found.

GT+IS: In order to improve previous solutions, we trim the graph again 32 units around the H2+IC and H12+GT solutions and solve the PCTLP with a time limit of 14400-t_1−t_2 seconds, where t_1 and t_2 stand for the time consumed by H2+IC and H12+GT,

Table 4
Computational results for random instances.

d	s	o	#Vars	#Ctrs	MIPGAP	Dev_UB				CPU Time			
					EX	EX	H2+IC	H12+GT	GT+IS	EX	H2+IC	H12+GT	GT+IS
17	5	5	171174	143535	25.0	0.2	1.5	3.7	0.0	TL	42.6	949.1	TL
		10	170565	143055	27.3	0.0	1.6	7.2	0.0	TL	47.7	947.6	TL
		15	170052	142650	26.8	0.0	1.6	10.2	0.4	TL	42.8	946.8	TL
	8	5	171174	143535	26.7	0.7	6.5	15.2	0.9	TL	65.5	970.3	TL
		10	170565	143055	27.0	2.6	5.8	12.9	0.3	TL	64.0	970.0	TL
		15	170052	142650	27.0	0.7	5.3	15.1	1.5	TL	63.5	971.4	TL
	12	5	171174	143535	27.1	1.7	6.9	15.8	0.0	TL	94.2	930.2	TL
		10	170565	143055	26.9	2.7	7.0	11.7	0.2	TL	105.3	1018.3	TL
		15	170052	142650	27.0	4.0	7.7	10.0	0.0	TL	92.7	1013.9	TL
17			**170597**	**143080**	**26.8**	**1.4**	**4.9**	**11.3**	**0.4**	**TL**	**68.7**	**968.6**	**TL**
33	5	5	1270624	1062214	37.7	6.4	0.5	15.2	0.0	TL	1956.0	1617.0	TL
		10	1266961	1059258	36.9	4.9	0.4	10.4	0.0	TL	1068.6	1614.9	TL
		15	1263660	1056593	36.9	4.9	0.7	12.6	0.0	TL	3030.2	1618.6	TL
	8	5	1270624	1062214	36.9	14.2	2.2	18.0	0.0	TL	1312.4	1947.2	TL
		10	1266961	1059258	36.7	13.4	2.2	20.7	0.0	TL	1073.6	1963.1	TL
		15	1263660	1056593	36.6	11.5	1.0	17.7	0.0	TL	1206.4	2041.3	TL
	12	5	1270624	1062214	36.6	8.8	2.1	24.2	0.0	TL	1620.8	2636.0	TL
		10	1266961	1059258	36.7	10.6	3.7	22.4	0.0	TL	1565.2	2556.4	TL
		15	1263660	1056593	36.6	7.0	3.0	20.6	0.4	TL	1864.5	2748.0	TL
33			**1267082**	**1059355**	**36.9**	**9.1**	**1.8**	**18.0**	**0.0**	**TL**	**1633.1**	**2082.5**	**TL**
Total			**718839**	**601217**	**31.8**	**5.2**	**3.3**	**14.6**	**0.2**	**TL**	**850.9**	**1525.6**	**TL**

respectively. Here we adopt as initial solution (IS) the best result from both H2+IC and H12+GT methods.

A time limit of 4 h (14400 s) was fixed for solving each instance with the exact approach (EX).

In Table 4, we report the average results of this first battery of computational experiments. The first three columns indicate the parameters identifying the instances, d (density), s (number of services), and o (number of obstacles). For each of the combinations of these parameters five instances were generated and we report the average results of the five instances. Column `#Vars` indicates the number of variables of the (MCMNFP) problem and column `#Ctrs` the number of constraints. Column `MIPGAP` indicates the relative MIP gap obtained at the end of the time limit for the exact method (EX). In the block of columns denoted by `Dev_UB`, we report, for each one of the procedures, the relative percent deviation of the obtained solution with that procedure (upper bound), z_{ub}, with respect the best solution among the two approaches, exact and GT+IS, z_{best}, that is:

$$\texttt{Dev\_UB} = \frac{z_{\text{ub}} - z_{\text{best}}}{z_{\text{best}}} \times 100.$$

Finally, in the block of columns `CPU Time`, we report the CPU time, in seconds, required by each one of the approaches. The flag `TL` indicates that the time limit was reached in all the instances averaged in that row.

A first analysis of the results shows that the exact approach was not able to solve, up to optimality, any of the instances within the time limit, being the `MIPGAP` significantly large. In addition, the average number of explored nodes in the branch-and-bound tree was below 10 for $d = 17$ and equal to 1 for $d = 33$ which implies that the `MIPGAP` is close to the gap with respect to the linear relaxation for those instances. This behavior was expected given the already known big size of the formulations in terms of variables and constraints. Regarding columns `Dev_UB` for EX and GT+IS, we can compare the results obtained by the exact method with the results obtained by matheuristic GT+IS in the same computational time (14400 s). We observe that the best solution (deviation from the best solution = 0.0) is obtained by GT+IS approach in most of the tested instances, especially for $d = 33$. Therefore, we can assert that matheuristic GT+IS outperforms the exact method.

On the other hand, the results on the deviations obtained by the heuristic approaches H2+IC and H12+GT show that H2+IC obtains better solutions in less computational time, being in many instances better than the solution obtained by the exact procedure, especially for $d = 33$. H12+GT also provides good results in small running times which also benefits significantly the results obtained with GT+IS.

Fig. 5 shows an example of a random instance $(d, s, o) = (33, 12, 15)$ together with the best obtained solution obtained with the matheuristic GT+IS. There, the squares (□) represent the sources, the crosses (×) the sinks, the obstacles are the light blue cubes, and the services are highlighted in `blue` (type A) and in `green` (type B). As can be observed, the solution shows that there are seven trays that combine four green and three blue services simplifying the design of twelve one-by-one single pipelines.

5.2. Corridor instances

In our second battery of experiments, we generate new instances encouraging the formation of cable trays in real situations. The set of random instances used in Section 5.1 are cubes containing random obstacles that may not favor the trays' formation. In order to force services to be grouped in trays, we have designed a specific set of instances with elongated shapes and specific obstacles layout. This collection of corridor instances is generated as follows.

The domain of this new set of instances is the 3D box $[0, 128] \times [0, 128] \times [0, 1024]$. An orthogonal 3D grid of $d_1 \times d_2 \times d_3$ physical nodes, with $(d_1, d_2, d_3) \in \{(9, 9, 65), (17, 17, 129)\}$ is built on this box, being the distance between adjacent nodes fixed to $\frac{128}{d_1 - 1}$ in X-axis, $\frac{128}{d_2 - 1}$ in Y-axis and $\frac{1024}{d_3 - 1}$ in Z-axis. This means a spacing of 16 and 8 units for $d = (d_1, d_2, d_3) = (9, 9, 65)$ and $d = (d_1, d_2, d_3) = (17, 17, 129)$, respectively. Here, d represents, again, the *density* of the grid. The graph for a corridor instance with density $d = (d_1, d_2, d_3)$ has $d_1 d_2 (d_3 - 1) + d_1 (d_2 - 1) d_3 + (d_1 - 1) d_2 d_3$ physical edges and $3 d_1 d_2 d_3$ virtual edges.

We consider a total of thirteen services partitioned into two types of services. There are eight services of type A that have to be sent from four sources, each one with an outgoing flow of two to four sinks, two of them with an in-flow of three, and another two with an in-flow of one. The remaining five services are of type B and depart from two sources with an outgoing flow of two and three, respectively to five sinks, each one with an in-flow of one. Two different scenarios, S_1 and S_2, are generated with these characteristics by randomly distributing sources

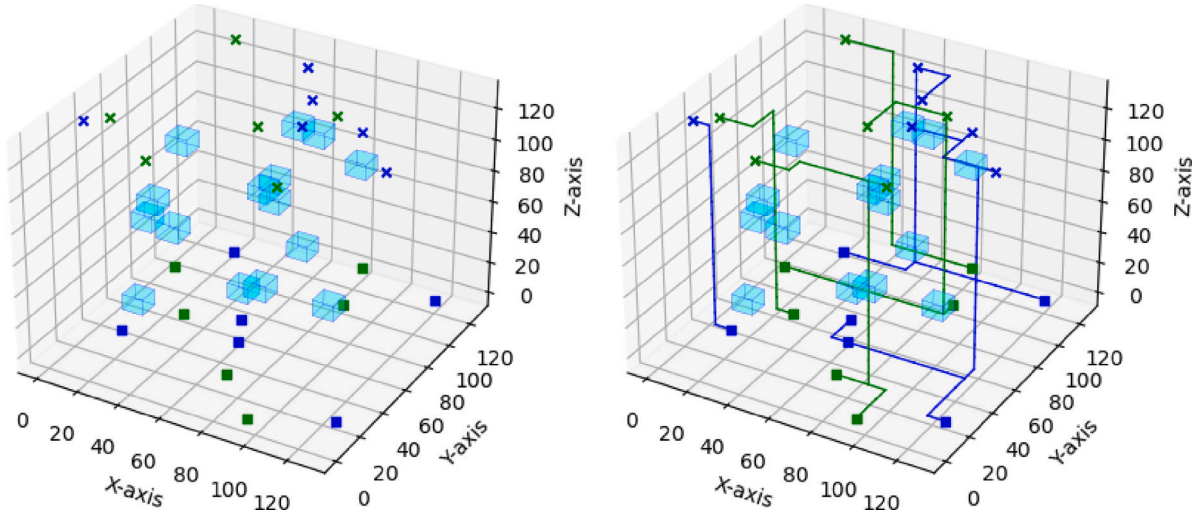

Fig. 5. Graphical display of a random instance $(d, s, o) = (33, 12, 15)$ and the best provided solution.

Table 5
Summary of the corridor instances.

Density	Service types		Obstacles			Radii
(9,9,65) (17,17,129)	Type A (8 services): 4 sources: 4 sinks: Type B (5 services): 2 sources: 5 sinks:	(out-flows: 2, 2, 2, 2) (in-flows: 3, 3, 1, 1) (out-flows: 2, 3) (int-flows: 1, 1, 1, 1, 1)	2 obstacles: O_{11} = {(16; 16; 288); (48; 128; 736)} O_{12} = {(96; 16; 288); (128; 128; 736)} 4 obstacles: O_{21} = {(0; 0; 288); (48; 48; 736)} O_{22} = {(80; 0; 288); (128; 48; 736)} O_{23} = {(0; 80; 288); (48; 128; 736)} O_{24} = {(80; 80; 288); (128; 128; 736)}			2

and sinks of the two types of services on the walls of the constructed 3D grid.

Two different groups of obstacles are also generated. The first group has two obstacles and the second one has four obstacles. Each obstacle is a 3D box that is given by their opposite vertices. In Table 5, we specify the vertices of obstacles as well as the different parameters that affect the generation of this new set of instances. In total, eight (2 densities × 2 scenarios × 2 groups of obstacles) corridor instances have been generated. Finally, we consider that each tray has enough capacity to gather all services of the same type and an established radius of $R^t = 2$ for all $e \in E$, $t \in \mathcal{T}$.

We consider the same cost structure as in the random instances in the previous section.

Fig. 6 shows two examples of these types of instances and the type of solution obtained. In these two examples, the squares (□) are the sources and the crosses (×) the sinks, in color `blue` for service type A and in `green` for service type B. In the left figure, two obstacles are considered and the sources and sinks are placed under scenario S_1. In the right figure, the instance consists of four obstacles, and the positions of the sinks and sources are generated under scenario S_2. The density in both figures is $d = (d_1, d_2, d_3) = (9, 9, 65)$. The reader should observe that Fig. 6 shows how these corridor instances favor the aggregation of services into trays routed by the feasible corridors defined by the obstacles.

In Table 6, we report the results obtained for these instances, with the same structure that Table 4 where column S indicates the scenario type. We observe that, again, the exact approach did not solve to optimality any of the instances, and that the MIP gaps at the end of the time limit are large. If we compare the exact method EX with GT+IS approach, we observe that the best solution is obtained by matheuristic GT+IS in all the instances. In addition, regarding columns `MIPGAP` and `Dev_UB` for EX and $d = (17, 17, 129)$, we can assert that this solution is near to the optimal solution.

On the other hand, again the results on the deviations obtained by the heuristic approach H2+IC show that this approach is able to obtain a very good solution, which is much better than the solution obtained by the exact procedure for $d = (17, 17, 129)$, in considerably less computational time.

5.3. Case study

This section presents an application of the proposed methodology to a real-world-based instance provided by our partner Ghenova, a leading Naval Engineering company. This is an instance with a number of services of different types and a series of obstacles and corridors that compromise the available space for the design (see Fig. 7). The structure of the instance is as follows.

- The designated space for pipelines and cable trays system design is a three-dimensional rectangular prism representing a ship cabin. Its dimensions are 5732 units (X-axis), 2836 units (Y-axis), and 2013 units (Z-axis).
- A grid was created for the cabin by dividing each axis into segments of 100 units, resulting in an initial grid of 58 × 29 × 21 nodes. Virtual nodes and edges were introduced to generate the graph $\widetilde{G} = (V, E)$, consisting of 73,407 virtual nodes and 141,101 edges (virtual and physical).
- The cabin houses five types of services identified by numbers 0 to 4. The distribution of services includes three services of the first type, two services of the second and third types, one service of the fourth type, and six services of the fifth type. Each service type can be accommodated within a tray, and their

Table 6
Computational results for corridor instances.

d	o	S	#Vars	#Ctrs	MIPGAP	Dev_UB				CPU Time			
					EX	EX	H2+IC	H12+GT	GT+IS	EX	H2+IC	H12+GT	GT+IS
(9,9,65)	2	S_1	130242	110066	16.6	0.5	4.4	82.0	0.0	TL	52.7	962.4	TL
		S_2	130242	110066	24.2	1.9	6.4	49.7	0.0	TL	72.9	970.4	TL
	4	S_1	114834	97010	16.2	0.0	4.8	63.4	0.0	TL	2077.9	1997.4	TL
		S_2	114834	97010	13.5	1.0	4.1	50.9	0.0	TL	1087.9	2155.3	TL
(17,17,129)	2	S_1	1000236	839210	60.2	78.4	3.3	78.4	0.0	TL	873.9	4138.4	TL
		S_2	1000236	839210	56.1	52.7	4.5	140.8	0.0	TL	950.0	2144.8	TL
	4	S_1	910698	763786	56.2	64.2	4.7	67.8	0.0	TL	655.5	1792.3	TL
		S_2	910698	763786	64.4	85.6	5.4	122.2	0.0	TL	655.5	1792.3	TL
Total			**910698**	**763786**	**38.4**	**35.5**	**4.7**	**81.9**	**0.0**	**TL**	**655.5**	**1792.3**	**TL**

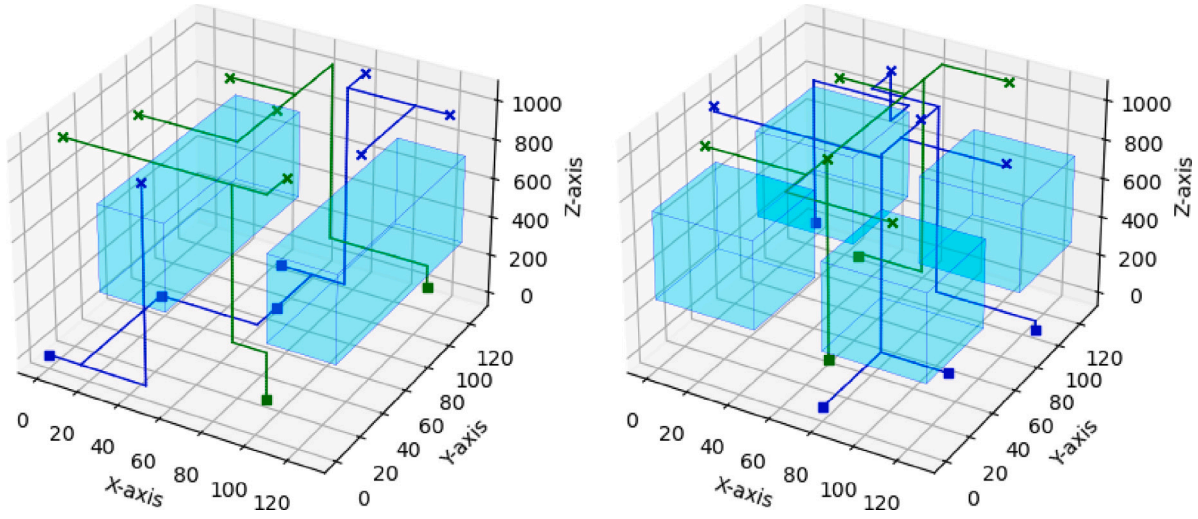

Fig. 6. Graphical display of the solution for two corridor instances.

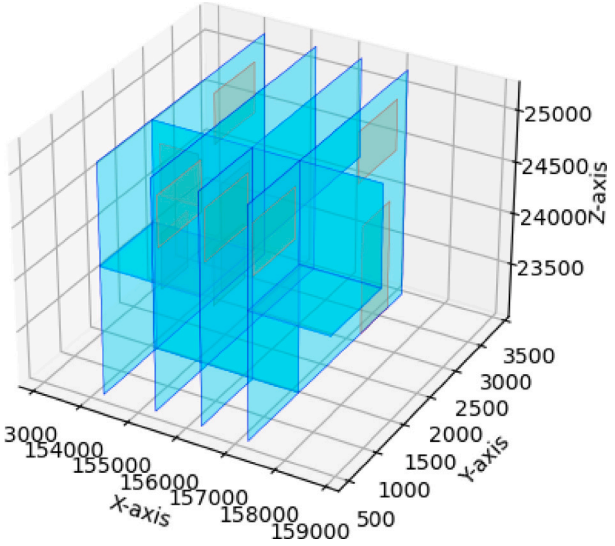

Fig. 7. Scenario of the case study.

sources and destinations are positioned throughout the cabin as per the designer's requirements.

- The routes for the trays face five obstacles (walls) depicted as light blue shapes in the plot. These obstacles represent metal slices with a width of 10 units and contain eight holes/windows represented by light orange squares, allowing passage.
- The designer specifies a minimum distance of 50 units between consecutive elbows, which is a typical requirement for this scenario.
- Each service type has assigned a different width: $R^1 = 50$, $R^2 = 25$, $R^3 = 75$, $R^4 = 150$, and $R^5 = 25$ units.

Once the grid is generated, we assign a cost to each of its edges, facilitating the evaluation of feasible routes within the network. The cost system incorporates the designer's preferences when routing the pipelines. It employs the following additive cost structure,

$$c_e^k = (\alpha_1^t + \alpha_5^t Pr_e)d_e + \alpha_2^t El_e + \alpha_3^k H_e + \alpha_4^t Ch_e + \alpha_6^t Pc_e + \alpha_7^t Cl_e,$$

considering the use of edges and elbows in each service's route. The cost calculation involves various parameters $(\alpha_1, \ldots, \alpha_7)$ influenced by the edge characteristics. Detailed criteria and parameters defining the cost system are provided in Table 7. This flexible cost structure aligns with the preferences of naval designers and has been determined based on the selection criteria expressed by our partner company.

The cost associated with each edge in the graph $\widetilde{G}$ is the same for all service types. In other words, $c_e^t = c_e^{t'}$ for any t and t' belonging to the set $\mathcal{T}$. The chosen parameters for this instance are:

α_1	α_2	α_3	α_4	α_5	α_6	α_7
1	2800	700	200	−0.7	4000	3000

Furthermore, two preference zones are defined as parallelepipeds with the following dimensions: $[154241, 158844] \times [535, 1419] \times [24249, 25221]$ and $[152013, 154241] \times [535, 3371] \times [24249, 25221]$.

We replicate in this case study the solution methods applied in the corridor and random instances described in Sections 5.1 and 5.2.

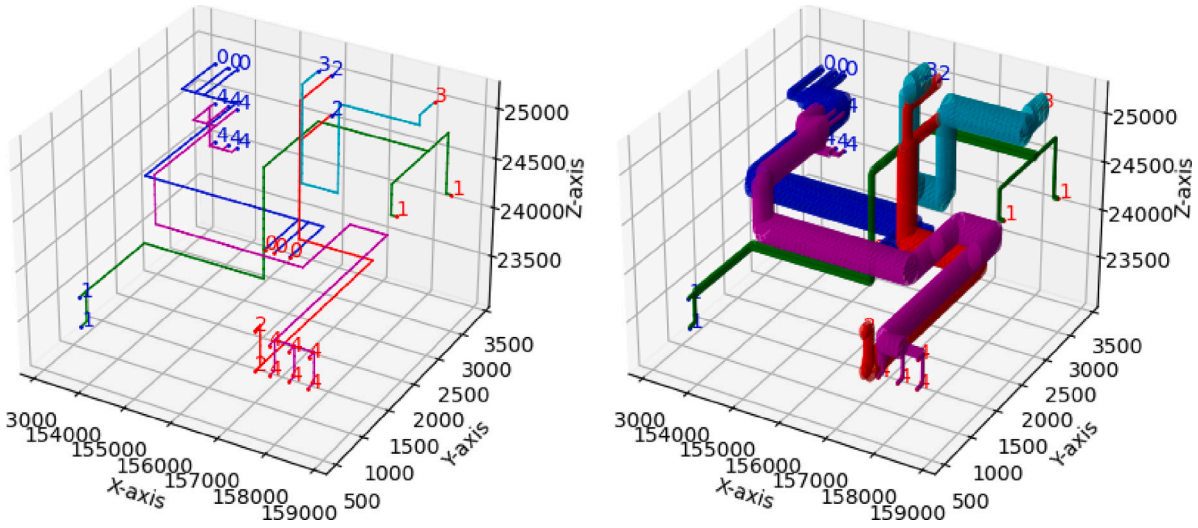

Fig. 8. Graphical display of the solution for the case study with 14 services. In the left it is shown the solution as a collection of edges. The figure on the right shows the same solution in 3-dimensions.

Table 7
Criteria and parameters involving the cost function of the case study.

Criteria	Description
d_e	physical distance between nodes i and j with $e = \{i, j\}$ (length of the edge).
E_e	1 if e represents an elbow and 0 otherwise.
$H_e = H - h_e$	being H the maximum height of the ship cabin and h_e the height of e in case it is in a plane parallel to the XY-plane and 0 otherwise.
Ch_e	1 if e is a vertical edge (in the Z-axis) and 0 otherwise.
Pr_e	1 if e belongs to a preference zone and 0 otherwise.
Pc_e	1 if e crosses a penetrable zone and 0 otherwise.
Cl_e	1 if e represents an elbow and it is close to a source or a destination point and 0 otherwise.
Parameters	**Description**
α_1^k	Cost per unit length.
α_2^k	Cost of an elbow.
α_3^k	Cost of moving away from the ceiling.
α_4^k	Cost of changing in z-coordinates (height). Therefore, moving to a different height is penalized.
α_5^k	Bonus per routing the pipeline in a preference zone ($\alpha_5^k < 0$; $\alpha_1^k + \alpha_5^k > 0$).
α_6^k	Cost of crossing a penetrable zone.
α_7^k	Cost of locating an elbow close to the source or destination points of a pipeline.

The reader may observe that the exact approach (EX) was not able to return a feasible solution after 4 h of computing time. This is shown in Table 8 with the flag NF. The best obtained objective function value is 323766.5, which is given by the GT+IS approach in four hours. We draw in Fig. 8 the obtained solution in the graph (left) and the 3D representation of the obtained solution as actual pipelines and trays (right). Each type of service is highlighted with a different color to ease the identification of the different types of services. We would like to highlight that for all types of services, the model favors the integration of several services (of the same type) in single trays. Additionally, our approaches assure that the obtained solutions satisfy the requirements on the minimum distance between consecutive elbows, the minimum distance between different types of services, and that the design is routed throughout feasible space avoiding forbidden regions defined by obstacles.

Matheuristic GT+IS produces, in view of the result obtained, a good quality solution in reasonable CPU time, providing the naval designer with a powerful decision-aid tool for this task. The results on the deviations and CPU times for H2+IC and H12+GT approaches appear in Table 8. These two approaches are less time-consuming and they are also able to obtain a very good solution.

6. Conclusions

In this paper, we provide different mathematical optimization-based tools to solve the Pipelines and Cable Trays Location Problem (PCTLP), a challenging problem of determining the routes of different pipelines and cable trays in an intricate obstacle-ridden space of a ship. Apart from the constructability conditions required for the routes (such as the separation of services, or the distance between elbows), some services are desired to be jointly routed through the same *cable trays*. We propose an exact relax-and-cut approach for solving the problem based on a min-cost multicommodity flow formulation. The large dimension of the formulation for medium-sized instances makes it advisable to consider some heuristic algorithms that provide good-quality solutions in smaller computing times. Thus, we have developed two families of heuristic algorithms and tested all our methods (exact and heuristic) in two batteries of computational experiments, namely randomly generated and of corridor type. In all cases, the exact method is only able to prove optimality for small- to medium-sized instances. However, for large-sized instances, it fails to prove optimality and, in some cases, cannot find any feasible solution within the time limit. This limitation underscores the model's challenge in handling larger problem sizes. The heuristic approaches, on the other hand, provide satisfactory solutions within a reasonable CPU time. However, they still face difficulties in achieving good results for large-sized instances within an acceptable time frame. To enhance solutions for these instances, it would be beneficial to explore other heuristic approaches to combine with our math-heuristic family to derive more effective solution algorithms. However, such an exploration exceeds the scope of this work. Furthermore, we conducted a real-world case study provided by our industrial partner to validate our proposal.

The methodology presented in the paper is rather general and admits the incorporation of new elements required in the design as additional constraints in the formulation and heuristics, as well as the integration of the PCTLP with other problems that appear in Naval Design. An important challenge is determining precise connection locations between pipelines or cables and the ship's power supplies, valves, and filters, following a predefined labeling scheme. The approval of a layout proposal depends on effectively addressing this connection

Table 8
Computational results for the case study.

	Dev_UB			CPU Time			
EX	H2+IC	H12+GT	GT+IS	EX	H2+IC	H12+GT	GT+IS
NF	3.0	0.6	0.0	14400.0	175.0	1689.0	14400.0

problem and adhering to the specified scheme. Key considerations include imposing topological restrictions on branch connections and strategically placing valves within the branches. After proposing a layout for the pipes and cables, the methodology involves identifying specific points where branches must connect to ship devices. An optimal routing problem is then solved to determine the final branch layout. Integrating the pipeline and cable tray layout problem with the connection problem enables the discovery of new solutions that would not arise from sequential approaches. This integration has the potential to reduce overall costs. Furthermore, another possible extension is integrating the pipeline and cable tray problem with ship cabin layout design (Wang et al., 2023). Although discrete optimization tools have been used to analyze these problems separately, a comprehensive joint study has not been conducted yet.

Declaration of competing interest

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

Data availability

We have shared the link to our data in the manuscript.

Acknowledgments

The authors of this research acknowledge financial support by the Spanish Ministerio de Ciencia *y* Tecnología, Agencia Estatal de Investigación, and Fondos Europeos de Desarrollo Regional (FEDER) via project PID2020-114594GB-C21. The authors also acknowledge partial support from projects: FEDER-US-1256951; Junta de Andalucía, Spain P18-FR-1422; CEI-3-FQM331; B-FQM-322-UGR20; AT 21_00032; NetmeetData: Ayudas Fundación BBVA a equipos de investigación científica 2019; Contratación de Personal Investigador Doctor (Convocatoria 2019) 43 Contratos Capital Humano Línea 2. Paidi 2020, supported by the European Social Fund and Junta de Andalucía; UE-NextGenerationEU (ayudas de movilidad para la recualificación del profesorado universitario); VII PPIT-US (Ayudas Estancias Breves, Modalidad A); and the IMAG-Maria de Maeztu grant CEX2020-001105-M /AEI /10.13039/501100011033.

References

Ando, Y., Kimura, H., 2011. An automatic piping algorithm including elbows and bends. In: International Conference on Computer Applications in Shipbuilding (ICCAS), Triestre, Italy. pp. 153–158.
Asmara, A., 2013. (Ph.D. thesis). Delft University of Technology, Netherlands.
Asmara, A., Nienhuis, U., 2006. Automatic piping system in ship. In: International Conference on Computer and IT Application. (COMPIT), Citeseer, pp. 269–280.
Blanco, V., González, G., Hinojosa, Y., Ponce, D., Pozo, M.A., Puerto, J., 2022. Network flow based approaches for the pipelines routing problem in naval design. Omega 111, 102659.
Castorani, V., Cicconi, P., Mandolini, M., Vita, A., Germani, M., 2018. A method for the cost optimization of industrial electrical routings. Comput.-Aided Des. Appl. 15, 747–756.
Dijkstra, E.W., 1959. A note on two problems in connexion with graphs. Numer. Math. 1, 269–271.
Dong, Z.R., Bian, X.Y., Zhao, S., 2022. Ship pipe route design using improved multi-objective ant colony optimization. Ocean Eng. 258, 111789.
Garg, M., Smith, J., 2008. Models and algorithms for the design of survivable multicommodity flow networks with general failure scenarios. Omega 36, 1057–1071.
Guimarães, J.P., Prata, B.d. A., 2022. Variable fixing heuristics for the capacitated multicommodity network flow problem with multiple transport lines, a heterogeneous fleet and time windows. Transp. Lett. 14, 84–93.
Guirardello, R., Swaney, R.E., 2005. Optimization of process plant layout with pipe routing. Comput. Chem. Eng. 30, 99–114.
Hightower, D.W., 1969. A solution to line-routing problems on the continuous plane. In: Proceedings of the 6th annual Design Automation Conference. pp. 1–24.
Jiang, W.Y., Lin, Y., Chen, M., Yu, Y.Y., 2015. A co-evolutionary improved multi-ant colony optimization for ship multiple and branch pipe route design. Ocean Eng. 102, 63–70.
Kim, S.H., Ruy, W.S., Jang, B.S., 2013. The development of a practical pipe auto-routing system in a shipbuilding cad environment using network optimization. Int. J. Naval Archit. Ocean Eng. 5, 468–477.
Lee, C.Y., 1961. An algorithm for path connections and its applications. IEEE Trans. Electron. Comput. 10, 346–365.
Orlin, J.B., 1997. A polynomial time primal network simplex algorithm for minimum cost flows. Math. Program. 78, 109–129.
Park, M., Jeong, B., Kim, M., 2020. Decision-making for cable routing at detailed ship design through life cycle and cost assessment. J. Int. Marit. Saf., Environ. Aff., Shipp. 4, 93–107.
Park, J.H., Storch, R.L., 2002. Pipe-routing algorithm development: case study of a ship engine room design. Expert Syst. Appl. 23, 299–309.
Pogorelskiy, S., Kocsis, I., 2022. Automation for structured cabling system in data centers using building information modelling. Int. Rev. Appl. Sci. Eng..
Rourke, P.W., 1975. (Ph.D. thesis). Lehigh University.
Salimifard, K., Bigharaz, S., 2022. The multicommodity network flow problem: state of the art classification, applications, and solution methods. Oper. Res. 22, 1–47.
Wang, Z., Yang, X.Q., Zheng, Y.H., Chen, W.C., Lv, P., Zhou, B., Xu, M.L., 2023. Interactive ship cabin layout optimization. Ocean Eng. 270, 113647.