Operations Research Forum (2023) 4:35
https://doi.org/10.1007/s43069-023-00208-5

ORIGINAL RESEARCH

Literature Survey on Automatic Pipe Routing

M. Blokland[1,2] · R. D. van der Mei[1,2] · J. F. J. Pruyn[3] · J. Berkhout[2]

Received: 14 September 2022 / Accepted: 20 February 2023 / Published online: 14 April 2023

Abstract
Piping systems are common in many architectures and designing such systems is often a complex task. Design automation of piping systems is therefore a universal research subject. Nonetheless, these piping systems are often still designed by hand as a result of their complexity. Consequently, costs associated with piping design are high, especially for large-scale architectures like ships and chemical plants. The goal of automatic pipe routing is to reduce the design time and associated costs of a piping system by automating the routing of these pipes. This survey provides an overview of the current state of automatic pipe routing literature to assist researchers and practitioners to further the study of automatic pipe routing. This is done by pinpointing and explaining the most important obstacles that stand in the way of making a full-scale automatic pipe routing method. The barriers that are analyzed are related to both model representation and optimization complexity. Finally, a synthesis table of research papers on automatic pipe routing is provided based on the handling of the aforementioned barriers and other general features of automatic pipe routing methodology. The survey concludes by discussing directions for further research.

Keywords Pipe routing · Space modeling · Optimization · Survey

1 Introduction

Piping systems are essential in many architectures like buildings, ships and chemical plants to transport liquids and gases from one place to another. Pipe routing of complex systems is still predominantly performed by hand using schematics (Piping

✉ R. D. van der Mei
mei@cwi.nl

1 Stochastics Department, Dutch National Institute for Mathematics and Computer Sciences, Science Park, Amsterdam 1098 XG, Netherlands

2 Mathematics Department, Vrije Universiteit Amsterdam, De Boelelaan, Amsterdam 1081HV, Netherlands

3 Department of Maritime and Transport Technology, TU Delft, Leeghwaterstraat, Delft 2628CD, Netherlands

and Instrumentation Diagram, or P &ID) and 2D / 3D drawings in Computer-Aided Design (CAD), see [1]. Consequently, pipe routing requires more than 50% of the engineering hours in ship design [2] and 25% of the production costs in chemical plant design [3]. Some software procedures exist as support tools [4], but pipe routing is mostly performed with the expertise of a piping engineer by modeling them one by one. Therefore, it can happen that an ongoing design blocks the next pipe to be routed, which could mean that a big portion of the design has to be revised before a feasible route for the next pipe can be found. Because automating the pipe design process could significantly decrease the production time and cost of any project that requires a piping design, the field of automated pipe routing (APR) is important. Research on APR methods is therefore not a new concept, the conceptual idea of APR has been a research subject for over 50 years. Nonetheless, after many theoretical methods and case studies, pipe routing in practice is often still performed by hand.

There are several problems that arise in APR that make all-encompassing APR methods currently unattainable. One critical problem is the computational complexity of pipe routing. Because pipe routing problems are highly nonlinear and the number of possibilities is countless, it is difficult for computer programs to capitalize on patterns within the mathematical model. This becomes especially problematic with the large set of constraints that exist for pipe routing [5] and the (subjective) qualitative nature of objectives and constraints for pipe routing [6]. Nonetheless, with the exponential increase of computational power and inventions of advanced algorithms and machine learning methods, the idea of an adequate automatic pipe router becomes more and more plausible.

To the best of our knowledge, there is only one short survey from 2008 on APR in the open literature [7]. Therefore, this extensive survey is intended to provide a complete overview of APR literature to assist researchers and practitioners to further the study of APR methods. The purpose of this survey is divided into three main goals, these goals are to:

1. Use APR literature to pinpoint and explain the most important barriers that obstruct the implementation of APR methods.
2. Provide an overview and categorization of procedures that are implemented to overcome these barriers.
3. Provide a concise overview of APR research papers and their proposed methods with the use of a synthesis table.

For the last goal, the most important features of multiple APR research papers are captured in the synthesis table. This is done by providing general features of the papers and by using the categorization of procedures to classify how APR research papers handle important barriers. A concept model is introduced that outlines the fundamental design structure of APR methods. This concept model can be found in Fig. 1.

This concept model categorizes two different segments that are essential in APR methodology, the *optimization model* segment and the *solution algorithm*

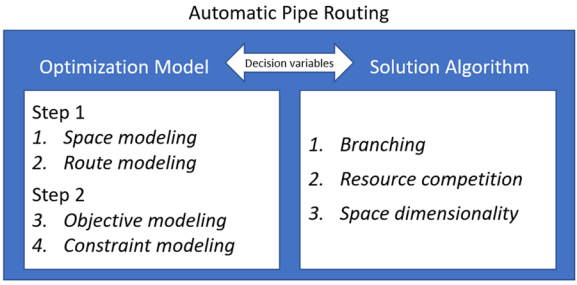


Fig. 1 A concept model of automatic pipe routing

segment. During the formulation of the optimization model, procedures are used to translate physical reality and design choices into a mathematical model. In the APR literature, procedures used in the formulation of the optimization model are commonly regarding four main components: *space modeling*, *route modeling*, *objective modeling* and *constraint modeling*. All these components are fundamental for the mathematical representation of the problem at hand. To model objectives and constraints, it is essential that some sort of specification is provided for the space and for the routes of pipes. One basic example is to check if a pipe collides with a piece of equipment. To check if this is the case, it is necessary to know both the location of the pipe and the location of this piece of equipment. Thus, space and route modeling is placed before objective and constraint modeling.

During the formulation of the solution algorithm, it is decided in which way the optimization model will be optimized. Note that the solution algorithm is highly dependent on the formulation of the optimization model. In the APR literature, procedures used in the solution algorithm are commonly regarding the following main components: *branching*, *resource competition* and *space dimensionality*. Branching is a component regarding the routing of a single pipeline among three or more connection points. Resource competition is about the competition of multiple pipes for limited resources, e.g., the room in cramped spaces. Space dimensionality is a barrier that is about the high quantity of options that results from the combination between multiple pipes and space complexity. Although branching is part of the formulation of the optimization model, it is commonly indirectly performed by treating a branched pipeline as a multitude of single pipelines. Consequently, the treatment of branching in the formulation of the optimization model is regarded as pipe route modeling and is further elaborated in Sect. 3.1.4.

The link between the optimization model and the solution algorithm is the set of decision variables, which are often the variables that describe pipe routes. The fundamental difference between the optimization model and the solution algorithm is in their relation to the decision variables. The optimization model determines the quality (objectives) and feasibility (constraints) of a piping design as explicit functions of a specific configuration of the decision variables. The solution algorithm determines how to search through the dimension of decision variables to find a sufficient (or optimal) configuration of decision variables. Note that the concept model does not imply a definitive chronology. Decisions for the formulation of the optimization model and the construction of the solution algorithm should be made jointly since are heavily dependent on each other.

This survey is structured as follows: In Sect. 2, the basic pipe routing problem is described. To achieve this, both a semantic and (fundamental) mathematical description are given for the pipe routing problem. Sections 3 and 4 explain and categorize the procedures that are used in APR literature to deal with the main components of the optimization model and solution algorithm, respectively. In Sect. 5, these categories are used to provide a synthesis table that compiles and summarizes all APR research papers used in this survey. Section 6 concludes the survey with a discussion on the current state of research in APR and possible future adaptations.

2 Automatic Pipe Routing Problem Description

Research on APR is often related to pipe routing in ship design [1, 2, 8, 9], aero-engine design [10–13], plant layout design [6, 14–16], cable routing [17–20] and water or gas distributing systems [21–24]. Nonetheless, there are many more domains where pipe routing is essential, for example, Mechanical, Electrical and Plumbing (MEP) engineering [25, 26], telecommunication satellites [27], electro-mechanical products [28], automobile underhood pipe assembly [29] and encapsulated oil pipes in turbines [30]. Although objectives and constraints differ significantly among several fields, a fundamental abstraction is present in all APR problems. This fundamental abstraction of APR can be defined as: *Using automated methods to outline capacitated routes within a confined space to connect multiple points such that predetermined requirements are achieved.* An example of an APR solution is shown in Fig. 2a, where a single pipe is routed between the two green blocks. The boundaries of the room in Fig. 2a indicate the boundaries of the confined space. The space within the confined space that is not occupied by obstacles is the space where can be routed, which will be referred to as the free space. The example in Fig. 2a can be showcased in 2D by an overhead view as shown in Fig. 2b. For convenience, 2D representations are used throughout this survey to explain new concepts as they provide a better overview.

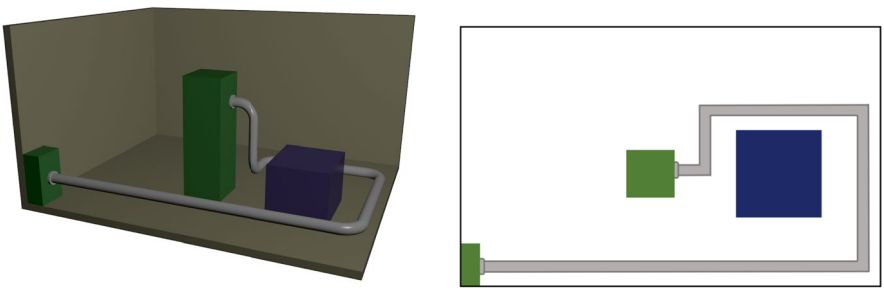


Fig. 2 An example that shows: **a** a 3D model of a APR solution, **b** a 2D representation of the 3D model shown in (**a**)

The term *requirements* within the definition for APR relates to the objectives and constraints associated with a specific problem. An example of a common APR objective is to minimize the length of the routes such as to minimize material cost. An example of an APR constraint is that pipe routes may not collide with solid obstacles. These obstacles can, for example, be electrical components or machinery. Even though “collision" with obstacles in the real world is not possible, these constraints are necessary because algorithms do not have these fundamental rules. The term *capacitated* in the definition for APR is used to indicate that the diameter of a pipe is incorporated in the physical constraints. This distinction is made because this is what makes pipe routing fundamentally different from standard routing problems. These diameter constraints can have a significant impact, especially in cramped spaces. This is illustrated in Fig. 3 by the use of a 2D representation of a pipe routing problem.

The difference between capacitated routing and non-capacitated routing is self-evident from Fig. 3. In essence, it means that the physical objects represented by the routes may not intersect. It can be seen that, although finding a route may be easy, the physical constraints introduced by the diameters can make a particular design state infeasible. This not only has an influence on the space between pipes but also on the space between pipes and obstacles. With the addition of the capacity (diameter) constraint, the pipe routing design problem transforms from a routing problem to a joint routing and packing problem. It should be noted that capacitated routing is a generalization of non-capacitated routing, as non-capacitated routing simply signifies that all pipes have a diameter of 0. Thus, even though this survey focuses on capacitated pipe routing, most methods can also be used for non-capacitated routing.

A few high-level mathematical descriptions of APR have been provided in previous research [6, 31–33]. The high-level theoretical idea of APR in this paper is built on the models of the aforementioned papers. Assume the following:

1. A metric $d : \mathbb{R}^3 \times \mathbb{R}^3 \rightarrow \mathbb{R}^+$.
2. The free space $\mathcal{F}$, which is a closed, bounded and path-connected subset of $\mathbb{R}^3$. The confined space $\mathcal{S}$ is then defined as the set of points within and on the boundary of the bounding rectangle of $\mathcal{F}$.

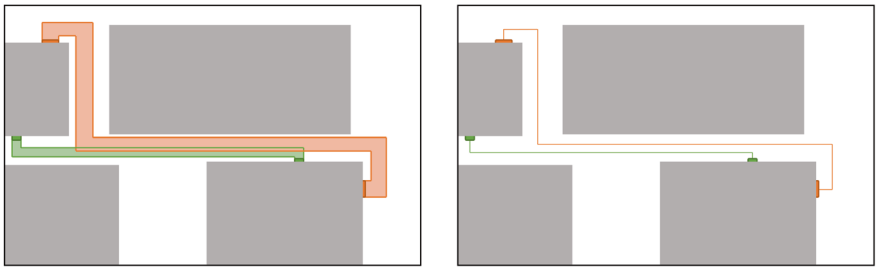


Fig. 3 An example that shows the difference between **a** non-capacitated and **b** capacitated pipe routing

3. The obstacle set $\mathcal{O}$ is a set of bounded, closed and path-connected sets called obstacles and is defined as follows: $\mathcal{O} = \{o_1, ..., o_m\}$ with $o_i \subseteq \mathcal{S}$ such that:

   lower-alpha o_i is a bounded, closed and path-connected set, $\forall i \in \{1, ..., m\}$.
   1 $o_i \cap o_j = \emptyset, \forall i \neq j,\ i,j \in \{1, ..., m\}$.
   2 $\bigcup_{i=1}^{n} o_i = \mathcal{S} \setminus Int(\mathcal{F})$, with $Int(\mathcal{F})$ the interior of the free space F.

4. $\mathcal{P} = \{p_1, ..., p_m\}$, a set of pipes to be routed within the free space, where $p_i = (C_i, d_i)$ with $C_i \subset \mathbb{R}^3$ a finite set of points in $\mathbb{R}^3$ that need to be connected and $d_i \in \mathbb{R}^+$ the diameter of pipe i.

Define the minimum distance of a point to a set of points as follows: $md[a, B] = \min_{b \in B}\ d(a, b)$. Now the pipe routing problem can then be explained as follows. For every pipe p_i, find a path-connected set of points (i.e., the route or centerline of a pipe) $\pi_i \subseteq \mathcal{F}$ such that:

1. $C_i \subseteq \pi_i$
2. $md[a, \pi_i] > d_i\ \forall a \notin \mathcal{F}$
3. $\nexists a \in \mathcal{F} : md[a, \pi_i] \leq d_i \wedge md[a, \pi_j] \leq d_j,\ \forall i \neq j$

Since this mathematical description is not something that can be directly solved by a computer, space and route modeling is used to approximate such problems, which will be discussed in Sect. 3.1. Note that this is a description of the fundamental idea of the pipe routing problem. Although this description may capture the parts of pipe routing that are always present in a pipe routing problem, there are other important case-dependent components that are not captured within this description. To accurately capture the idea of a specific APR problem, it is also necessary to look at the constraints and objectives of the problem. Constraint and objective modeling in APR literature will be discussed in Sects. 3.2.1 and 3.2.2, respectively.

As mentioned in Sect. 1, decision variables form the link between the optimization model and the solution algorithm. As APR methods are based on finding specific routes for all pipes, it is often the case that decision variables are equivalent to the route modeling parameters. A few different decision variable definitions exist in APR literature. One such method is to use the location of pipe supports as a decision variable, after which pipes are routed between these supports with a standard shape [34]. Nonetheless, decision variables are always closely related to pipe route representations.

The goal of the optimization model is to model space and routes in such a way that objectives and constraints can be adequately represented without the loss of routing potential. This means that the modeling of decision variables is regarded to be roughly synonymous with route modeling. Examples of decision variables used in APR literature can thus also be found in Sect. 3.1. Objectives and constraints are direct functions of the decision variables. It can be the case that these functions are not analytically tractable, methods to deal with such problems will be discussed in Sect. 4.4.

The optimization model provides a framework that can indicate the quality and feasibility of a problem given a specific configuration of the decision variables.

There are multiple components that can cause complexity in the optimization of the decision variables. The goal of the solution algorithm is to exploit problem structures to reduce the number of configurations of decision variables that have to be checked. This can be performed by using approximations and heuristics. How APR literature implements such procedures will be discussed in Sect. 4. Nonetheless, this survey will first examine the way in which formulations of APR optimization models are handled in literature in Sect. 3.

3 Optimization Model

As mentioned before, the formulation of the optimization model concerns the procedures used for the translation of physical reality and design choices to a mathematical model. To represent APR problems as a mathematical model, it is necessary to give specific descriptions of the routing space and the possible pipe routes in such a way that interaction between the two is possible. The mathematical description of APR provided in Sect. 1 is too abstract to simply solve on a computer, especially when objectives and constraints are modeled in the same way. To express the objectives and constraints as mathematical functions of the decision variables, it is necessary to model the space and routes as mathematical variables. Therefore, procedures that are used in the formulation of the optimization model first commit to space and route modeling, after which constraints and objectives can be expressed in terms of space and routes. Since space and route modeling are closely related, they will be examined simultaneously in Sect. 3.1. Constraint and objective modeling in APR literature will be examined in Sect. 3.2.

3.1 Space and Route Modeling

Space modeling is a necessary first step to express the objectives and constraints of the problem in a viable way. If a model is to capture every possibility of routing a pipe, it is required to use a continuous representation for these routes. Although such methods are used in APR occasionally (see Sect. 3.1.2), most APR research remodels these continuous problems using some kind of discretization method. This is done by either limiting some continuous variables to a discrete set of options or by making a complete discrete model of routing space by only allowing pipes to go through certain points in space, often resulting in the use of a graph. One example of a completely discrete model is a grid graph as shown in Fig. 4. Grid graphs are often used in APR literature because they restrict bends to 90 degrees and offer a framework in which objectives and constraints can be modeled with ease [5]. For such graphs, the routing space is divided into several uniform cells which can then be used to specify pipe routes. Figure 4 shows how such a grid graph is constructed. The APR problem in Fig. 4a is first discretized by decomposing the space into several uniform cells. Every grid cell then gets a certain value corresponding to its continuous counterpart as shown in Fig. 4b. In this example, the grid cell values can be: *free space* (white), *pipe 1* (green), *pipe 2* (orange) or *obstacle* (gray). To

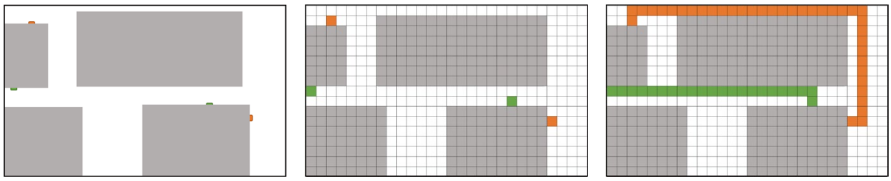


Fig. 4 An example that shows the specification of a grid graph

make a translation to a graph, every cell can be seen as a vertex having edges to every neighboring vertex. A pipe route can now be specified by finding a path on the imposed graph from the starting vertex to the ending vertex as shown in Fig. 4c. Objectives and constraints can also be expressed in functions dependent on edges and/or vertices. An example of an objective could be that the length of a route can be approximately minimized by using the minimum number of grid cells to go from start to end. An example of a constraint is to prohibit routing over obstacle grid cells to avoid collisions with obstacles. This is just one method to model space and routes but it shows how such a discretization method can make the modeling of objectives and constraints considerably easier.

Space modeling can be performed with different levels of detail. This level of detail is positively correlated with the complexity of the resulting problem that has to be solved. A one-to-one description of space and routes will often result in a more complex mathematical problem, while an easy-to-solve mathematical problem often reduces the actual problem to an abstract form. The idea of space and route modeling in APR is thus to find a satisfactory balance between complexity and path quality. One way to deal with this is to use multiple space representations with different levels of details, which will be discussed in Sect. 4.3. This section will focus on single space and route representations.

One way to simplify the pipe routing model is to use a taxicab geometry. A taxicab geometry is similar to a Euclidean geometry but uses a Manhattan distance metric, defined as: $d(A, B) = \sum_{i=1}^{Dim(A)} |A[i] - B[i]|$ [35]. Because the Manhattan distance measures the distance of the shortest orthogonal path between two points, the fundamental idea of a taxicab geometry is that only paths with 90-degree bends are allowed. Szykman and Cagan [36] name several reasons why this method is often preferred over a Euclidean geometry in APR: A taxicab geometry restricts sharp bends that can cause pressure drops. Additionally, it facilitates the parallel routing of pipes which results in installation operations becoming more practical and pipes becoming more accessible. Belov et al. [6] note that: “As done in the literature, we limit our pipe routing approach to rectilinear axis-parallel routing; in particular, we constrain all bends to be 90 degrees. This is acceptable as non-90 degree bends are extremely rare in real plants, ...”. Nonetheless, use may vary from field to field, in aero-engine design, non-orthogonal routes are adopted more frequently than in ship design [37]. Routing in a Taxicab geometry is referred to by many names, some of which are: orthogonal routing, Manhattan routing, rectilinear routing, lattice paths

and routing using Von Neumann neighborhoods. This survey will refer to such methods as orthogonal routing.

Literature on APR methods often uses discretization methods to reduce the problem to a well-known mathematical concept, the graph. Nonetheless, several research papers tackle APR in more detailed ways that do not discretize the space to the level of a graph. Some APR methods take a continuous approach where pipes can be routed through every point in space. The practical application of both discrete and continuous approaches will be discussed below, respectively. Section 3.1.3 provides an analysis of APR obstacle modeling in practice. Additionally, Sect. 3.1.4 elaborates on how branching is taken into account during the formulation of the optimization model.

3.1.1 Discrete Approaches

Graph methods use graph representations as abstractions of the free (routing) space. Route definitions are represented using the vertices and edges from this graph abstraction. Since graphs representations do not offer a lot of options in how routes are represented, differences are most often found in the way space is represented. The graph methods can be grouped into two distinct, but closely related, categories: cell decomposition methods (also known as navigation meshes) and skeletonization methods (also known as road maps). Cell decomposition methods refer to methods that divide the free space area into several non-overlapping areas called cells, where each vertex is related to a cell and edges are found between adjacent cells. One example of a cell decomposition method is provided in Fig. 4. Important to note for cell decomposition methods is that, in general, a path over multiple vertices does not give a precise route, as vertices are related to areas not to single points. Skeletonization methods refer to methods that relate the vertices of a graph to specific points in space. Edges of a skeletonization graph are continuous lines that connect these points in space. Both methods are showcased in Fig. 5.

In Fig. 5a, an example of a routing space is given, which is decomposed into rectangles in Fig. 5b. Figure 5c shows a cell decomposition graph of the space where the vertices of the graph are the light blue areas while the edges are the dark blue lines under the arrows. Figure 5d shows a skeletonization graph of the same space. Here, the vertices are exact points in space represented by the blue circles and the edges are the gray lines, which are continuous lines through the free space that connect these points. While the resulting graph may be the same, their inherent properties are different. A part of research on APR, especially in the domain of water and gas distributing systems, assumes that space representation in form of a graph is provided beforehand and just focuses on the optimization of the graph itself.

Note that cell decomposition methods and skeletonization methods are closely related. A single point in every cell can be chosen after which a route can be found over every edge in the cell decomposition to get a skeletonization graph. Of course, an infinite number of points can be chosen in every cell, in this way cell decomposition methods can be seen as generalizations of multiple skeletonization graphs. Skeletonization methods have a more precise route definition but limit the routing possibilities. On the contrary, cell decomposition methods keep route definitions vague

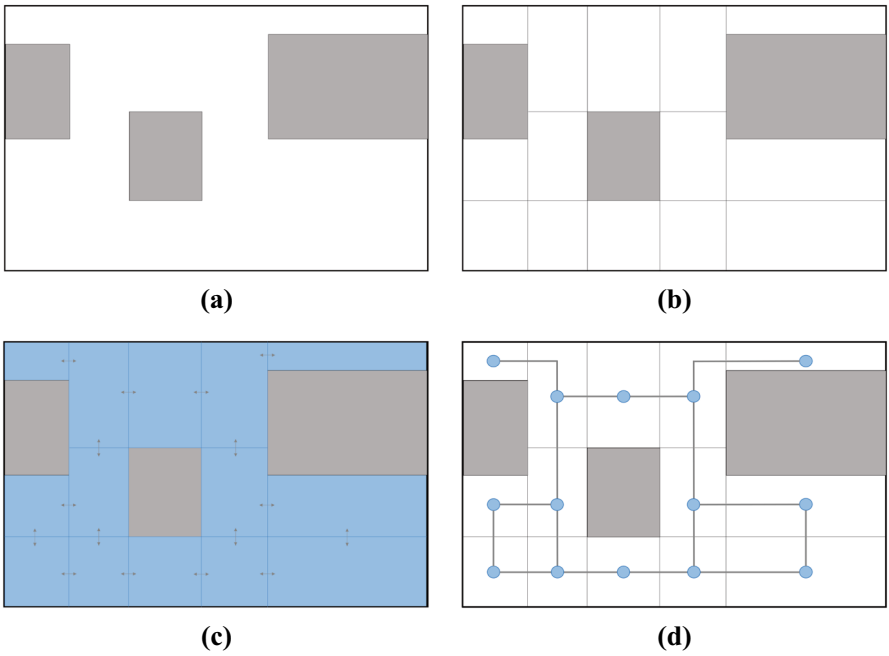


Fig. 5 An example that shows different methods for graph construction or confined space (**a**); **b** shows a decomposition of the confined space, **c** shows a cell decomposition graph while **d** shows a skeletonization approach

such that multiple routing possibilities are still left open. As mentioned before, 2D examples are used for easy interpretation but most methods can be directly extended to 3D. This is not always the case however, one example is given by Dielissen and Kaldewaij [38], who show that the computational complexity of making a minimum rectangular cell decomposition of a confined space without detached obstacles is polynomial in 2D but $\mathcal{NP}$-Complete in 3D. One typical method to deal with the translation from 2D to 3D is to use a finite number of 2D planes to approach a 3D routing space [8, 39, 40]. Additionally, in aero-engine design, a cylindrical representation is sometimes used [11, 37, 41, 42], which can also make the translation from 2D to 3D complicated.

Figure 6 provides several different cell decomposition methods that are regularly used in APR literature. Figure 6a shows a uniform grid, which is a widely used graph method in APR literature [1, 5, 43]. This particular method can impose orthogonal routing by only allowing up/down and left/right movements in a cell. There is a choice to be made on the width of the grid cells, which in research is often related to the diameter of the smallest pipe to be routed [8, 44]. Figure 6b shows a quadtree, which subdivides a cell into more cells when details increase in obstacles. Such a method is called an octree in 3D and is sometimes used for APR methods [11]. Figure 6c shows a non-uniform grid used in [31, 45, 46], that is constructed by drawing orthogonal lines from every obstacle point until the boundary of the routing space is reached. Figure 6d shows a trapezoidal decomposition in which either only

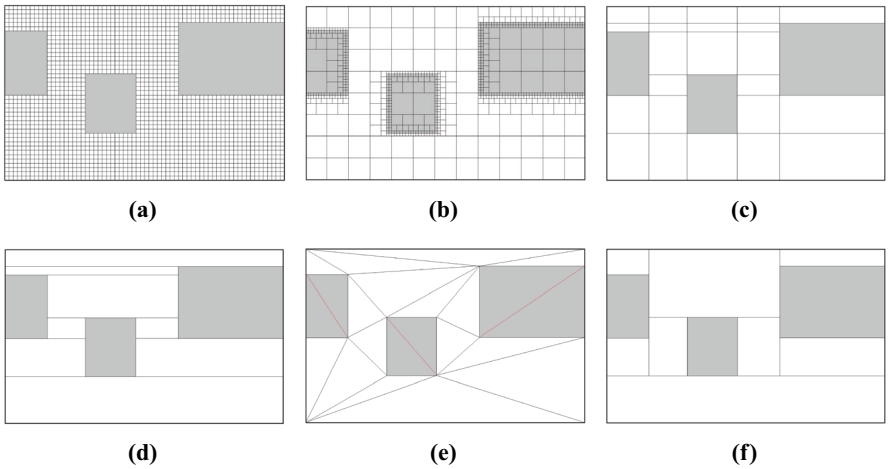


Fig. 6 An example that shows different methods for cell decomposition; **a** a uniform grid cell decomposition, **b** a quadtree cell decomposition, **c** a non-uniform grid cell decomposition, **d** a trapezoidal decomposition, **e** a Delauney triangulation and **f** a minimum orthogonal decomposition

horizontal or only vertical lines are drawn from obstacle corner points, used in [47], this is also called corner stitching [48, 49]. Figure 6e shows Delauney triangulation, used in [27, 50]. Figure 6f shows a minimum convex or orthogonal decomposition. Although this last method has not yet been adopted in APR literature to our knowledge, it is important to note that research on cell decomposition methods exists, in which certain aspects (like the number of cells or minimum edge lengths) are minimized [51]. Every cell decomposition method shown above can also be used as a skeletonization method, this is done by relating every line of the cell decomposition to an edge, and relating every point where more than two lines come together to a vertex. For example, the lines of the non-uniform grid cell decomposition method (Fig. 6a) can be used as a skeletonization graph [15, 52]. The same can be done for uniform grid cell decomposition methods [15, 31], which is also called a lattice graph or a mesh graph.

Figure 7 provides several different skeletonization methods. Figure 7a shows a Hanan grid, which is a skeletonization method of the specific non-uniform grid graph in Fig. 6c. Figure 7c shows an escape graph used in [10, 53]. The escape graph is closely related to the Hanan grid but instead of drawing orthogonal lines from every obstacle point to the boundary of the routing space, they are drawn to the first obstacle that is reached. There are also specific skeletonization methods that are not directly related to cell decomposition methods. Figure 7b shows a visibility graph, used in [54, 55]. The visibility graph has the property that the shortest (non-orthogonal) path in a routing space with polygon obstacles lies on this specific graph, for more information see [56]. Figure 7d shows a generalized Voronoi diagram, which specifies for every obstacle, the region of free space that is closest to this obstacle. The lines between these regions are then the collection of points that have the same distances to the (at least) two closest obstacles, see [57] for more

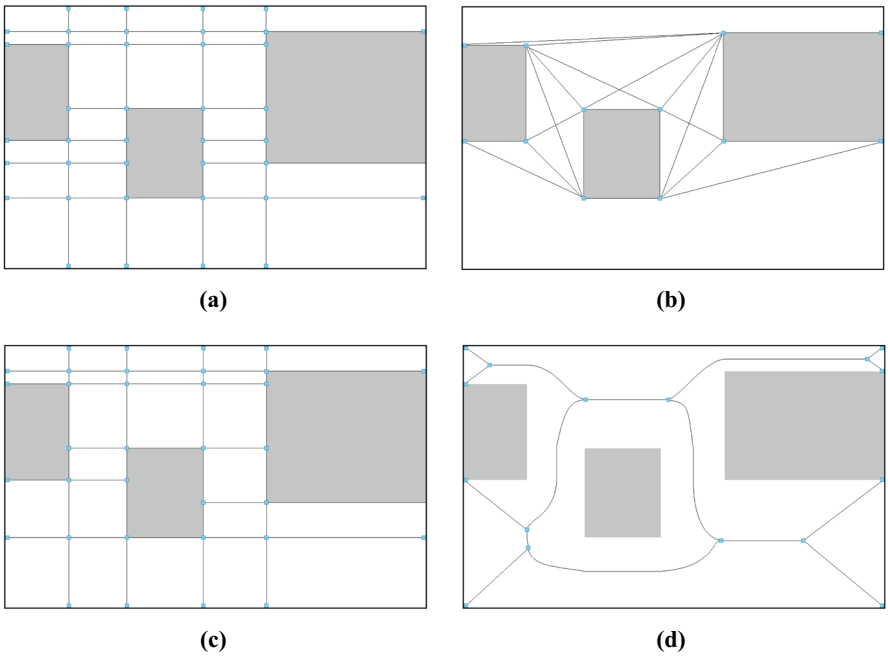


Fig. 7 An example that shows different methods for cell decomposition; **a** a Hanan grid, **b** an escape graph, **c** a visibility graph and **d** a generalized Voronoi diagram

information on generalized Voronoi diagrams. Besides these more widely used skeletonization methods, some new methods are also introduced in APR literature. One example is the track graph [53]. A track graph closely resembles an escape graph (Fig. 7c), but reduces the number of vertices in the graph. Lui [58] introduces the Manhattan visibility graph, which is an orthogonal visibility graph that resembles an escape graph but also vastly reduces the number of vertices. Yuan et al. [37] propose a compressed visibility graph that limits the number of vertices of the visibility graph. Thantulage et al. [59] uses random-based sampling for vertex placement to construct a skeletonization graph. Another original example is the chaos grid preprocessing model [28], in which a logistic map is used to map grid points and create a graph. Although this survey focuses on methods used in APR literature, many more methods for graph creation can be found in the literature on robotics and game development [60].

Note that even though most discrete approaches in APR literature use a graph, there are some papers that have different discrete approaches for space and route representation. These methods often use a discrete set of basic shapes to go from one place to another [2, 3, 44]. For such methods, pipe routes have only a finite number of possible basic styles (minimum Manhattan distance and bends) to go from one point to another. Another method is provided by Shin et al. [61], in which routes are seen as sphere-shaped agents. These agents can move with a constant step length in six possible directions in 3D (up, down, left, right, forward and backward).

Table 1 Table of continuous specificity that is implemented in previous automatic pipe routing research

Source	Length	Bend	Source	Length	Bend
[26]	Cont.	Disc.	[62]	Cont.	Cont.
[63]	Cont.	Cont.	[40]	Cont.	Cont.
[12]	Cont.	Cont.	[34]	Cont.	Disc.
[64]	Cont.	Cont.	[36]	Cont.	Cont.
[50]	Cont.	Cont.	[30]	Cont.	Cont.
[29]	Cont.	Both	[27]	Cont.	Disc.
[59]	Both	Both	[65]	Both	Both
[13]	Cont.	Cont.			

3.1.2 Continuous Approaches

Besides the graph-based approaches in the previous section, there are also some approaches that find routes for pipes without the restrictions imposed by discretization. To provide an accurate model for APR, a specification of both space and routes has to be provided. Since pipe routes should be able to visit any point in free space for the continuous approach, it is sufficient to model just the obstacles and the boundaries of the confined space for collision detection. Pipe routes can be fully explained by just two distinct sets: the set of its straight pipe segments and the set of its bends. As a result, pipe routes in APR literature are generally expressed by a combination of these two sets. Bends are often described by their angles and straight pipe lengths by their length. This survey uses these two properties to explain the level of continuous specificity of a pipe routing representation. These properties are called *length-continuity* and *bend-continuity*. Length-continuous pipe routing implies that the length of the straight pipe segments can be chosen without limitation, while bend-continuous pipe routing implies that the angle of a bend can be chosen without limitation. All graph methods shown before are both length-discrete and bend-discrete, as both are limited by a discrete set of possibilities. Note that there are more possible continuous characteristics that are not included, as these are not discussed in APR literature. One example is the shape of the bends, which is often pre-specified and constant beside the bend angle. Limiting bend shapes also limits the routing possibilities.

As can be seen in Table 1, most methods that incorporate continuous pipe lengths also incorporate continuous bend angles. This is often the case because a continuous approach for both can be implemented without much complexity. The general method that is used for continuous APR in literature is performed as follows: Routes are represented by a diameter and an ordered list of points in space. These points in space represent bend points and the straight lines between every pair of consecutive bend points denote the centerline of the straight pipe segments. This is depicted in Fig. 8. In this way, the only decision variables that are needed to optimize the problem are the number of bends and the bend locations. The implied centerlines/routes of a pipe and its diameter can then be used for collision detection and other objectives and constraints.

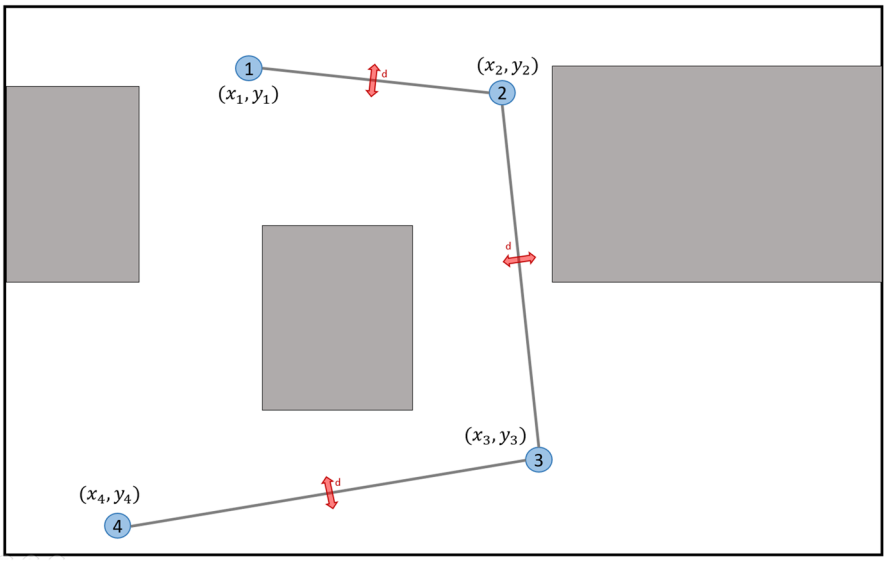


Fig. 8 An example of a continuous pipe representation, here (x_i, y_i) represents the location of point i and d is the diameter of the pipe

Although this basic modeling method is most often used for continuous pipe routing, there are other methods for modeling space and routes in a continuous way. Sandurkar and Chen [29] describe pipe routes by direction cosines, straight pipe lengths and bend angles. In this way, they can use both discrete and continuous approaches to describe bend angles. Wang et al. [26] make use of continuous insertion points which are required to be visited but restrict bends to 90 degrees. In this way, pipes are expressed by multiple points in space and standard orthogonal shapes (basic styles) to connect these points.

3.1.3 Obstacle Modeling

Since obstacle modeling for discrete approaches is performed before discretization, the obstacle modeling methods used for discrete and continuous approaches are often similar. One common example is that obstacles are often increased by pipe radii such that a collision-free route can be defined by its centerline [1, 32, 44, 63]. Sometimes, an extra distance is added to take minimum clearance constraints into account [11, 15]. Both discrete and continuous approaches share the desired properties that: 1. Obstacle representations should not be too complex. 2. Obstacle models should be able to closely resemble physical reality. The property that obstacle models should resemble physical reality means that virtual obstacles can be used to adequately model constraints and objectives imposed by physical reality. The two properties above are often conflicting; obstacle models that closely resemble physical reality are generally complex while simple models often do not closely resemble physical reality. In APR literature, obstacles are regularly represented as orthogonal

shapes using minimum bounding box techniques [2, 16, 37, 66], or as a sum of several orthogonal shapes [8, 46, 67]. Asmara et al. [68] introduces the optimized subdivision boundary box method, which analyzes the obstacles to make a fitting virtual obstacle using the sum of orthogonal shapes (boxes in 3D). This method is later used by Niu et al. [69] and Asmara [1].

Nonetheless, there is a difference in obstacle modeling between discrete and continuous approaches that should be noted. Discrete approaches often require very basic (orthogonal) descriptions for one of two reasons: 1. The more detailed the obstacles are in a routing space, the more cells are needed for a proper cell decomposition. 2. The more detailed the obstacles are in the routing space, the more obstacle points they have and thus the more vertices skeletonization methods will have. Since continuous approaches are often solved using metaheuristics (see Sect. 4.4), they can use more elaborate and detailed obstacle descriptions. The reason for this is that metaheuristic algorithms do not directly use the information structure contained in constraint and objective functions. This means that objectives and constraints can be represented by complex, intractable functions like specialized collision detection algorithms. In such cases, complex obstacle representations can be used, for example, tessellated object representations [29, 59, 65].

3.1.4 Branched Pipe Route Modeling

Although pipe routing with branches is distinctly different from pipe routing without branches, it often does not have a considerable impact on how the optimization model is constructed in practice. A single pipe with branches is commonly treated as a set of multiple pipes that have to be routed such that they are all connected [39, 64, 70]. Consequently, pipe routes for branched pipes are then simply defined as a collection of (non-branched) pipe routes, all having their own objectives and constraints. Note that there is a multitude of methods to route these sub-pipes and to assure that they are linked. Nonetheless, since these methods are integral to the solution algorithm and are not part of the formulation of the optimization model itself, they will be discussed in Sect. 4.1.

3.2 Objective and Constraint Modeling

In this paper, the meaning of the words *objectives* and *constraints* is similar to that in mathematical optimization problems. An objective can be seen as an explicit goal with the purpose of being optimized, expressed as a scalar, and measured by a function of the decision variables. A constraint can be characterized as a limitation on the possibilities of decision variables. Decision variables are a collection of settings that precisely describe the routes of all pipes[1]. The goal of APR is to find a set of decision variables such that all constraints are realized and the objectives are optimized to an adequate level. Since multi-pipe routing problems are often

[1] Assuming the mathematical description of APR as described in Sect. 2, the decision variables describe the collection of all continuous lines $\{\pi_i\}\ \forall i \in \{1, ..., m\}$.

integer, highly nonlinear problems, a small change in the decision variables can have a substantial impact on the quality and/or feasibility of a certain design state. This implies that, for every pipe routing problem with a different set of objectives and constraints, it is needed to (re-)analyze these objectives and constraints to gain information on that specific pipe routing problem. Section 4.4 provides an overview of how specific algorithms can capitalize on different information structures within the objectives and constraints.

Note that not every objective or constraint mentioned below is present in all APR problems, or that the objectives and constraints mentioned cover all possible objectives and constraints. There are many types of pipe routing problems all with their own characteristics, this can also be seen in the concise APR literature overview provided in Sect. 5. The goal of this section is to provide the reader with a detailed account of different constraint and objective modeling methods that are used in APR literature. Note that this survey may classify some constraints and objectives in another category than in the source that is referenced. For example, some papers classify pipe length as a construction (installation) cost. Since this survey categorizes this as a material cost, it is regarded as such.

3.2.1 Constraint Modeling

The goal of this section is to provide the reader with a detailed overview and categorization of constraint modeling methods that are used in APR literature. APR constraints are sometimes separated in soft and hard constraints [1, 23, 61] or (having the same meaning respectively) in restrictive and quantifiable constraints [2, 7, 31, 44]. Hard or restrictive constraints imply that they must be satisfied at all times, soft or quantifiable constraints imply that they can be disregarded when this shows to be either advantageous or necessary. As mentioned before, this survey treats constraints as limitations on decision variables that must be satisfied at all times. Thus, every constraint is a hard constraint. Soft constraints are seen as objectives because these are measures that should be optimized and do not directly limit the decision variables.

APR constraints are often divided into shape constraints and location constraints [28, 33, 71]. Shape constraints are constraints that put limitations on the shape of the pipe. Location constraints are constraints that put limitations on the locations where pipes can be routed. A common method to model location constraints is to use virtual obstacles and virtual sinks [16, 33, 55, 63]. Virtual obstacles do not exist in physical reality but are modeled as such to restrict certain areas. Virtual sinks are used to attract certain pipes to preferred areas. While the distinction between location and shape constraints is a useful distinction to make, this survey will provide a more detailed categorization of constraints. The different types of constraint categories are: *physical constraints*, *safety constraints*, *operational constraints* and *maintenance constraints*[2]. These categories will be explained independently below.

[2] Note that it frequently happens that some restrictions are advantageous for multiple types of constraints. For example, a substantial part of operational and maintenance constraints overlap, as operational infeasibility often coincides with an increase in repair and maintenance times. To avoid repeating such constraints as much as possible, they are assigned to the most applicable category.

Physical Constraints There are several different kinds of physical constraints that are discussed in APR literature. Physical constraints are constraints that make sure that a set of decision variables do not lead to a physically infeasible piping design. These constraints frequently consist of collision avoidance constraints and pipe shape constraints. These two categories and associated approaches in APR literature are given below.

- Collision avoidance constraints: As mentioned before, obstacles are often enlarged with the radius of the pipe to be routed. Ikehira et al. [67] increase pipe diameters by the size of its accompanying flanges to avoid collisions between flanges and other physical objects. Enlarging obstacles for physical constraints can also be done dynamically for pipes with heterogeneous diameters. Dong and Bian [8], for example, introduce temporary obstacle grid cells that can be changed to normal grid cells when pipes with different diameters are routed. Although avoiding obstacle collisions is always necessary, it is sometimes allowed during optimization to prevent the algorithm from getting stuck in local optima. This is commonly done with the use of potential energy [14, 25], which is a way to grade grid cells based on their location (also see Sect. 3.2.2). Allowing temporary collisions for global optimization can also be done with the use of other penalty functions, see, for example, [29, 30, 66, 67]. One such method is introduced by Sandurkar and Chen [29], in which the number of tessellated triangles that are crossed by pipes is minimized. To make sure these kinds of methods always avoid collisions when the algorithm is finished, the weight of the penalties can be increased over iterations in the algorithm [23]. Pipe collisions are often also avoided by routing pipes sequentially, see, for example, [6, 29, 69]. In this way, pipes that have already been routed can be treated as obstacles. Nonetheless, this does have an effect on the optimization complexity. As this is part of the *resource competition* component of the solution algorithm, it will be discussed in the associated section (Sect. 4.2).
- Pipe shape constraints: One simple pipe shape constraint is that pipes are commonly routed in an orthogonal way, examples are [1, 5, 26]. Occasionally, the straight pipe segments between bends are restricted to have a minimum length [27, 72]. Such restrictions are implemented as they can be imposed by suppliers [27]. Qu et al. [41] extends pipe nozzles coming from obstacles with a certain length such that the directional orientation of the pipe nozzles is taken into account.

Safety Constraints Safety constraints are constraints that make sure that every design state is safe. Pipe routing safety is extremely important, especially for aero-engines. Ren et al. [42] note that: "General Electric (GE) Company investigated the airplane accidents caused by sudden stops of aero-engines and finally was shocked by the conclusive finding that about 50% of these kinds of accidents occurred due to the invalidation of pipes, wires and/or sensors." Safety constraints are often incorporated by the use of restricted areas or minimum distances from specific equipment or machinery. Most of them can be grouped into the following categories: virtual obstacles, minimum distances and potential energy. These categories and associated approaches in APR literature are given below.

- Virtual obstacles are often used to make sure pipes do not cross unsafe areas [46, 55, 63, 73]. This can be done for all pipes or for a subset of pipes (e.g., no heat-sensitive pipes near combustion engine [61]). How these virtual obstacles are modeled depends on how obstacles are modeled. One example is to make certain vertices in a graph unavailable [46].
- Minimum distances are often used to ensure pipes are not routed too close to certain areas to make sure a design state is safe [5, 71, 74].
- Just like for obstacle avoidance, potential energy can be used for safety constraints to allow certain transgressions during optimization to avoid local optima [8, 10, 41, 53].

Operational Constraints Operational constraints are constraints that attempt to exclude all design states that adversely impact operational procedures. Most operational constraints can be grouped into the following categories: hydraulic feasibility, thermal expansion and accessibility necessities. These categories and associated approaches in APR literature are given below.

- Hydraulic feasibility: Hydraulic feasibility constraints are constraints regularly used in the APR domain of gas and water distributing systems, see, for example, [21, 23, 75, 76]. Nonetheless, they are also applied in other APR domains [15, 27, 63, 71]. Although flow and pressure are highly dependent, research does not always take both into account at the same time. Flow constraints relate to the number of bends, slope, drop limits, and pipe diameters [15, 21, 23, 27, 63, 71]. Pressure constraints relate to pipe lengths, pipe diameters, number of bends and bend angles [15, 24, 36, 63, 75]. These pressure and flow constraints can be implemented using associated methods, for example, the Hazen-Williams equation [22, 23] or with the use of Fluid-Structure Interaction (FSI) [63].
- Thermal expansion: Thermal expansion constraints are constraints that prevent excessive stress levels caused by thermal expansion. Belov et al. [6] use a guided cantilever method for flexibility relating to thermal expansion. Kumar and Cheng [15] make sure thermal stress does not exceed a given limit.
- Accessibility necessities: It is often the case that virtual obstacles are introduced to prevent routing within operational areas [47, 77]. One example is to simply remove vertices from the graph that lay in operational places [78]. Zhu and Latombe [33] take valve accessibility into account by restricting the height of associated pipes such that valves can be reached by hand.

Maintenance Constraints Maintenance constraints are constraints that attempt to exclude all design states that adversely impact maintenance procedures. Most maintenance constraints can be grouped into the following categories: stability & reliability and accessibility necessities. These categories and associated approaches in APR literature are given below.

- Stability & reliability: Supports are often used for stability and reliability in APR literature. Stanczak et al. [27] attach pipes to floors or walls in an orthogonal fash-

ion for stability reasons. Kumar and Cheng [15] take into account both sustained and occasional stress levels of pipes by using supports. Dong and Lin [44] avoid pipe paths with air pockets.

- Accessibility necessities: As with operational constraints, maintenance accessibility constraints often use virtual obstacles to block routing within maintenance areas, see, for example, [16, 55, 63, 74]. One example is that pipes should not be routed in front of equipment or close to the engine [42]. Another example is to route pipes that require frequent maintenance on the outer layers of the aero-engine such that they are easily accessible [63].

3.2.2 Objective Modeling

The goal of this section is to provide the reader with a detailed overview and categorization of objective modeling methods that are used in APR literature. The most common method to model objectives is with the use of potential energy. Potential energy for grids was introduced by Ito [43] and is used by most APR methods that include a grid. Even some continuous approaches use a grid for the sole purpose of incorporating potential energy [26]. To take heterogeneous pipes into account, this potential energy can be either dependent on the type of pipe [25] or more generally, dependent on the diameter of the pipe [41]. The reason why potential energy (and therefore a grid cell decomposition method) is so widely used in APR literature is that it can be used to model many different types of spatial objectives. Asmara [1] shows that the simple idea of potential energy can be used for: routing points, obstruction areas, sink areas, rough areas, attraction areas, magnet areas, distraction areas and special-type areas. As with constraint modeling, objectives used in APR literature can be categorized into several different categories. These categories are: *material cost objectives*, *construction objectives*, *operational objectives*, *maintenance objectives* and *aesthetic objectives*. It is often the case that objectives within these groups are expressed in the same quantitative or qualitative manner but this is not always the case[3]. These categories will be explained independently below.

Material Cost Objectives Material objectives are about the direct costs of pipes. Material costs are often kept simple by focusing solely on the total routed length of the pipes, see, for example, [20, 46, 55]. Even when other objectives are implemented, the length of pipes is commonly the most important objective of APR. Nonetheless, there are some other variables related to material costs that are frequently used. One such variable is the diameter of pipes [2, 15, 23]. These diameters can be modeled as continuous variables [22, 79] and as discrete variables [24, 52]. Although having a set

[3] It can be difficult to find a function that accurately represents certain subjective and qualitative goals, this is especially the case for complex issues like valve operability or design aesthetics. Objectives can also have fundamentally different evaluation methods, making an objective ordering between them impossible. Such objectives often clash in optimization, such that improvement in one objective leads to a deterioration in another objective. Optimization of such a set of incomparable objectives is called multi-objective optimization and is often handled by either one of two methods; 1. Creating a single scale for all objectives assuming a certain subjective trade-off among these objectives. 2. Using Pareto-efficiency for optimization. More on multi-objective optimization and Pareto-efficiency can be found in [135].

of discrete pipe diameters can complicate optimization, this decision is made since suppliers often only have a finite set of diameters [24]. Note that the number and angle of pipe bends could be considered material costs when they are bought from suppliers. This survey will list these under construction costs because most research on APR categorizes pipe bends as construction costs under the assumption that bent pipes are made, not supplied.

Construction Objectives Constructions objectives are used to make the construction/installation of the pipes as cheap and smooth as possible. Pipe bending has to be performed by a person on a bending machine, the number and angle of bends have specific man-hours costs. Therefore, minimizing pipe bending costs is a common objective. Additionally, objectives for reducing support costs are also frequently taken into account. Most construction objectives can be grouped into the following categories: pipe bends, parallel routing and support structures. These categories and associated approaches in APR literature are given below.

- Pipe bends: The cost of pipe bending can be different between two distinct types of pipe. Park and Storch [2] note that "The bending cost follows a step function because cold bending is used for small pipes and high-frequency bending is used for large pipes. The bigger the pipeline diameter, the more costly the bending, because it takes more man-hours." There are several methods that quantify the number and angle of bends. As shown in the quote above by Park and Storch, they take into account different types of bending costs dependent on pipe diameters. Sui and Niu [80] reduce the number of bends while backtracking using the Maze algorithm. Sometimes a maximum number of bends is implemented [27, 54, 59, 81]. This can be done to either limit the number of bends but also to make sure Genetic Algorithms can be modeled using a fixed-length encoding (see Sect. 4.4). Other methods that are used are: minimizing average bend angles [13] and using a discrete set of bend angles (i.e., a finite bend catalog) [62].
- Parallel routing: For construction purposes, it is often the case that pipes should be routed close to each other such that they can use the same pipe supports. Potential energy can also be used to route pipes in parallel [39, 44, 71]. Implementation of this method can be done by giving higher potential energy values to grid cells that lay next to an already routed pipe [39]. In this way, newly routed pipes will prioritize routes along pipes that already have been routed. Besides potential energy, other methods have been implemented to achieve routing pipes in parallel. Some methods are: Minimizing the lengths of supports [64]. Route pipes in parallel as a function of projection overlaps, distances and angles between pipes [37]. Directly optimize distances between pipes [8]. Use virtual sinks [1, 33] to guide currently routed pipes to pipes that have already been routed. Dong and Bian [8] prioritize parallel routing for pipes with equal properties, i.e., pipes with the same diameters or with the same pipe-dependent constraints.
- Support structures: Pipes are often routed as close to walls, floors, ceilings, equipment or machinery as possible such that no external support has to be used. Proposed methods generally use potential energy for routing along obstacles. This is implemented by giving grid cells close to obstacles higher potential energy.

Besides potential energy, other methods for routing close to obstacles are: Minimizing the average radial distance of discrete points to obstacles [37]. Requiring bends to be in a support zone and minimizing the cost of virtual support structures [6]. Using an evaluation function for the smallest perpendicular distance to obstacles [78] and minimizing the length of pipes that is not along walls [82].

Operational Objectives Operational objectives are objectives that are used to ease future operational procedures. Most operational objectives can be grouped into the following two categories: energy costs and accessibility. These two categories and associated approaches in APR literature are given below.

- Energy costs: Pipe diameters can be optimized for energy objectives after a piping design has been achieved [15]. Nonetheless, several methods jointly optimize pipe routing and energy objectives. Shiono et al. [24] use a mixed integer nonlinear programming to optimize pipe length and pipe diameters such that customers receive the right pressure gas. Pressure and heat drops are often directly modeled in the evaluation function [78, 79]. One way to do this is to minimize pipe lengths, number of bends and bend shapes for pressure drops [15, 37]. To take flow objectives into account, penalty functions can be used for nodal head and pipe flow velocities [21, 22]. Energy loss can also be prevented by analyzing how thermal insulation layers can be used [79].
- Accessibility: Operational accessibility objectives are often used to minimize the discomfort caused by pipes during operational procedures. Potential energy can be used to limit pipe routing inside operational areas [8]. Wu et al. [83] use fuzzy functions as a penalty for traversing areas meant for machine operation. Ikehira and Kimura [84] and Kimura [9] optimize valve operability by using three levels for valve accessibility: good, fair and bad. The level of accessibility is calculated with a recursive fill algorithm that mimics the movement of a person, after which the sum of minimum distances from each valve to accessible segments is minimized. Park and Storch [2] use a combination of valve operation frequency, manhour costs and a coefficient that signifies difficulties and discomfort levels to quantify valve operability.

Maintenance Objectives Maintenance and repair objectives are objectives that are used to ease future maintenance and repair procedures. Most of these objectives can be grouped into the two following categories: stability & reliability and accessibility. These two categories and associated approaches in APR literature are given below.

- Stability & reliability: In aero-engine design, it is often the case the pipes are routed as close to the surface as possible for stability and reliability reasons [42, 55]. Afshar [21] models the reliability of the piping system as the number of independent paths that can secure flows when unforeseen failures happen.
- Accessibility: As with many objectives, maintenance accessibility objectives are often quantified by potential energy. Maintenance spaces can be predetermined, after which potential energy can be used to avoid these spaces [26, 41, 85]. Besides potential energy, punish functions can be used for crossing main-

tenance and repair areas [42, 67]. Another method can be used when pipes are routed sequentially. The idea is to route pipes that require frequent maintenance and inspection last, since they will be blocked less often by previously routed pipes [40].

Aesthetic Objectives Aesthetic objectives often coincide with other objectives. One example of such a shared solution is to route pipes orthogonal. While this may be beneficial for aesthetic reasons, it is often performed with some other objective in mind. Nonetheless, several aesthetic objectives and methods of implementation are specifically mentioned in APR literature. One such method is to route pipes in bundles as much as possible [86]. Another method is to arrange pipes to have the same height as much as possible, this can be implemented by using different (grid) layers on the height dimension [39]. Qu et al. [41] note that "... pipes should be laid along the axial and circumferential directions of engines.", this is performed by giving priority to these routing directions. Potential energy can also be used to make sure that pipes do not interfere with other equipment [41].

4 Solution Algorithm

The formulation of the solution algorithm involves the procedures used to overcome obstacles that obstruct the optimization of the optimization model. The most important property of a problem with regard to optimization is its computational complexity. Therefore, this section will address several main components that introduce computational complexity. This section assumes that the reader is familiar with the essence of computational complexity theory. A detailed account of computational complexity can be found in [87].

It has been shown that the complexity of APR is an element of the $\mathcal{NP}$-Hard class [88], which practically means that the time it takes to find the optimal solution of the problem increases exponentially with respect to the problem size. APR is an element of the $\mathcal{NP}$-Hard class because there are several problems that are part of or equivalent to the pipe routing problem, which are shown to be in $\mathcal{NP}$-Hard. Some examples are: Numberlink [89], Circuit Design [90–92] Multi-Agent Path-finding [93] and Unsplittable Multicommodity Flow Problems [94]. Note that all these resembling cases are regarding the pipe routing problem in a setting that uses a graph, i.e., a discretized version of the actual problem. Nonetheless, for continuous representations, it is known that finding the shortest path of even a single pipe in 3D is $\mathcal{NP}$-Hard as it directly translates to a Euclidean shortest path problem [95]. This changes when obstacles can be fully described by obstacle points, for which a visibility graph can be used to find the shortest path in polynomial time [96]. For more information on Euclidean shortest paths, this paper refers to [97].

There are several components that contribute to the optimization complexity of an APR problem. The three most important barriers to the optimization of APR problems will be discussed in this survey. These three barriers are: *branching* (Sect. 4.1), *resource competition* (Sect. 4.2) and *space dimensionality* (Sect. 4.3). These barriers are chosen as they are caused by specific problems and thus, procedures to overcome

these barriers will share the same goal. Another reason why these three barriers are chosen is that variability among procedures to overcome these barriers is high in APR literature. As mentioned before, for every barrier, procedures proposed in the literature are categorized and explained such that they can be summarized in Sect. 5. Section 4.4 will analyze the advantages and disadvantages of several types of algorithms, as algorithms often have built-in methods for dealing with complexity from many different kinds of barriers.

4.1 Branching

The idea of branching is that single pipes need to be connected to three or more endpoints. This means that the pipe needs to have at least one point on its route where it is split into a branch. Although pipe branching is often not taken into account in APR methods, they are present in most pipe routing design problems. Asmara [1] notes on pipe routing in ship design that: "Most of the shortest path algorithms are basically only concerned with finding the shortest path that connects two points, while in practice, more than 70% of pipes have a branch." The idea of branched pipe routing is to minimize the total length of the pipe and its branches together. This is related to well-known problems in both graph theory, called Steiner minimal trees (SMT), and in geometry, called geometric Steiner minimal trees. In geometric Steiner minimal trees (GSMT), a distinction can be made between Euclidean minimum Steiner trees (EMST) and rectilinear minimum Steiner trees (RMST). The SMT problem is shown to be part of the complexity class $\mathcal{NP}$-Hard [98]. Additionally, the GSMT problem is shown to be part of the complexity class $\mathcal{NP}$-Complete [99], making both problems exceedingly hard to optimize.

Figure 9 gives an example to illustrate the concept of SMT. Figure 9a provides a graph with terminals and Steiner points, Fig. 9b provides the SMT of this graph. In an SMT, terminals are vertices that must be connected to each other and Steiner points are vertices that may be used to connect these terminals. An SMT is a collection of edges such that every terminal is connected and the sum of the edge weights (often related to distances) is minimized. Both an EMST and RMST have specific terminals that correspond to points in a geometric space. Any other point in this geometric space can then be seen as a Steiner point. The idea of a GSMT is to minimize the sum of the Euclidean (EMST) and Manhattan (RMST) distance while making sure the terminals are connected. Pipe branching in a discrete environment directly translates to the SMT problem and pipe branching in a continuous environment directly translates to the GSMT problem, making it a difficult variable to optimize in any space setting. Several methods have been proposed to approximate or find the exact minimum Steiner trees in relation to APR. These methods will be discussed and categorized in this section. Although several methods for the optimization of Steiner trees are explained below, research on this topic is extensive. See the survey of [98] for more information on the Steiner Minimum Tree problem.

As mentioned above, pipe branching in a continuous environment translates directly to a GSMT problem. Note that to translate the continuous APR problem into a GSMT problem, the obstacles also have to be taken into account. Such problems are

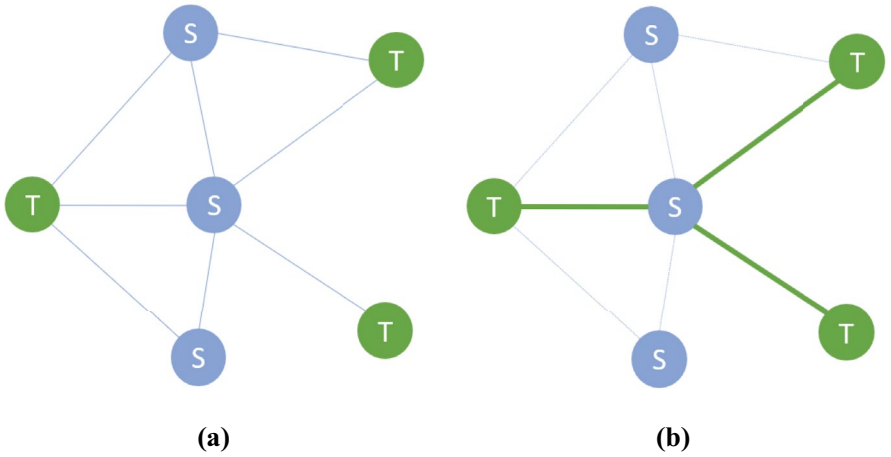
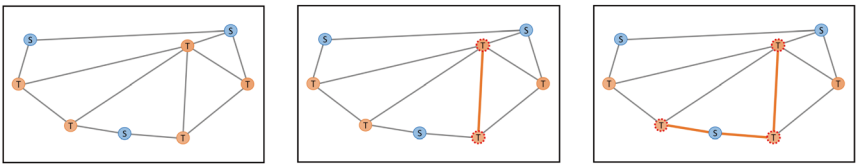


Fig. 9 An example that shows **a** a graph with terminal and Steiner points and **b** its associated minimum Steiner tree

often called Obstacle Avoiding Steiner Minimal Tree (OASMT) problems. Although such methods are not directly incorporated in APR literature, it is widely applied to similar problems. Hu et al. [100] translate a Circuit Design routing problem into an OARSMT (Obstacle-Avoiding Rectilinear Steiner Minimal Tree) problem. See the survey of [101] for more information on the OARSMT problem. Additionally, reducing (OA)RSMT problems to the boolean satisfiability problem (SAT) such that they can be used by state-of-the art solvers also seems to be advantageous for both the RSMT [102] and OARSMT [103] problems.

The start and endpoints of a branched pipe can have several different diameters, i.e., it is possible that a pipe branch has a smaller diameter than the actual main pipeline [2, 8, 69]. Asmara [1] makes the distinction between fixed branches and floating branches. It is noted that: "In a fixed branch there is a master pipe and a child pipe ... In a floating branch, all nozzles have the same priority." It is also noted that if branches are routed sequentially, the hierarchy in fixed branches can be used for determining the routing order of branches. This method is also adopted by Dong and Bian [8], Dong and Lin [44] and Niu et al. [69], who restrict branch connections of only one level difference in hierarchy[4]. Park and Storch [2] see branch pipelines as either end-forked (branches of a pipe that lie near one of the endpoints) or middle-forked (branches of a pipe that do not lie near one of the endpoints). They use different methods to deal with each kind of branch and also incorporate a recursive method that deals with multi-level branches. Ganley and Cohoon [104] show that an optimal rectilinear (orthogonal) Steiner minimum tree problem with obstacles can be constructed by only using the vertices in an escape graph. Finding an SMT on an escape

[4] Hierarchical branching indicates that there is a finite ordered set of pipe diameters. One hierarchical level difference between two branches implies that there is no pipe diameter that lies in between the diameters of these branches.



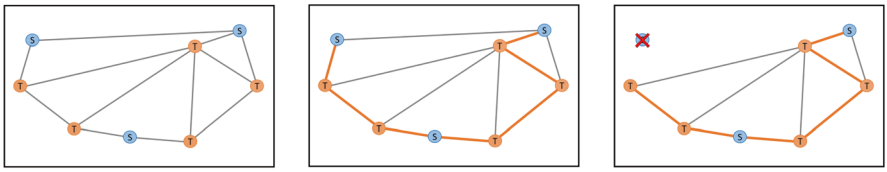
(a) (b) (c)

Fig. 10 An example of a sequential branching method

graph is thus equal to finding an RMST problem with obstacles in a taxicab geometry. This survey makes the distinction between four main groups of methods used for branching in APR literature: *sequential method*, *minimum spanning tree method*, *parallel method* and *exact method*. An overview of these methods is provided below.

Sequential Method Sequential branching methods treat branched pipe routing as a collection of single non-branched pipe routing problems. This is often done by initially creating a path between two terminals, after which new terminals are added in a sequential manner. Figure 10 shows an example of how the sequential branching method can be used to add terminal points one by one. The orange vertices are the terminal vertices that all have to be connected with each other. The blue vertices are the Steiner vertices, which can be used but are not required to be used, to connect the terminal vertices. This is a useful method for multi-level branching problems because main pipelines can be given priority by routing them first. Sequential branching can be performed by first finding a main path between two terminal points and then iteratively creating a new (branched) path by choosing a random terminal point and connecting it with the current path [1, 33, 39, 70]. Instead of choosing a random terminal point to route next, it can also be chosen to be the terminal point closest to the current path [105]. Some methods find a main pipeline after which branch points are placed on this pipeline to dictate where pipes should connect [9, 28]. Jiang et al. [64] break down the problem of branches by treating it as a collection of vertex pairs that resemble single pipes.

Minimum Spanning Tree Method These methods create a minimum spanning tree (MST) as an approximation for an SMT. Since an MST can be found in polynomial time, it can be used as a fast approximation for an SMT. Figure 11 shows an example of how such an approximation method can be implemented. First, an MST is used to connect all vertices, after which Steiner vertices can be removed from the tree until there are no Steiner points left that can be removed. Several MST branching methods are implemented in APR literature to reduce complexity and improve the solution. Wu et al. [79] use an iterative procedure of Kruskal's algorithm by adding vertices until the graph is connected. Some methods use the properties of visibility graphs and the removal and shifting of Steiner points to reduce the SMT problem to an MST problem [54, 66].



(a) (b) (c)

Fig. 11 An example of a minimum spanning tree branching method

Parallel Method Parallel branching methods iteratively look for a global optimum with the use of metaheuristic algorithms like Genetic Algorithms (GA), Particle Swarm Optimization (PSO), or Ant Colony Optimization (ACO). More on these metaheuristic algorithms can be found in Sect. 4.4. Branch pipe routing is often solved through ACO algorithms by letting ants in different colonies meet each other, see, for example, [11, 22, 75, 82]. Qu et al. [11] implements this idea with a modified max-min ant system to represent a pipe by a group of ant colonies that walk together and can split/branch out when needed. Dong and Lin [39] use the same cooperative system in the context of a PSO algorithm. Liu and Wang [10] use PSO, combined with the properties of an escape graph, to find an orthogonal minimum Steiner tree with obstacles for a single pipe. Savic and Walters [23] use a minimum spanning tree method as an initial solution for the graph Steiner tree method and improves this solution using a genetic algorithm. Jiang et al. [82] use a sequential branching method as an initial solution for their Co-evolutionary Improved Multi Ant Colony Optimization (CIMACO) algorithm, after which initial points are changed by the CIMACO algorithm to improve the global solution.

Exact Method These methods find an exact solution for the branch pipeline problem. One such example is provided by Shiono et al. [24], in which cyclic orderings are used for all the demand vertices in a graph as introduced by Bern [106].

4.2 Resource Competition

This component of the solution algorithm is concerned with multi-pipe routing in a space with limited resources. Multi-pipe routing implies that more than one pipe has to be routed in the same confined space. Limited resources indicate that a choice has to be made to decide which pipes can make use of these resources. One such example is the room that has to be shared by pipes, in which a routing order of pipes can be used to prioritize certain pipes. Pipes are commonly routed one-by-one, both in practice and in literature, see, for example, [58, 70, 77]. Since any pipe that has been routed will form an obstacle for the next pipe to be routed, the order in which pipes are routed can have a large influence on the final design. One example is shown in Fig. 12.

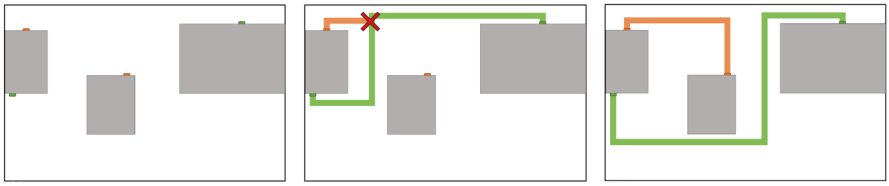


Fig. 12 An example that illustrates how routing order can impact routing feasibility

From the example in Fig. 12, it becomes clear that first finding the shortest route for the green pipe can cause feasibility problems for the orange pipe (Fig. 12b), when overlaps are not allowed. It is also clear that a feasible solution can be found by changing the routing order such that the shortest path for the orange pipe is found first (Fig. 12c). The limited resource here is the space between the three obstacles since both pipes would be shorter if they are routed optimally through this space. Every pipe can be seen as a dynamic obstacle, in which changes have an impact on the objective function but also on the constraints of all other pipes. This introduces significant complexity, as ordering n different pipes results in $n!$ permutations of unique pipe orders. This can also be seen when looking at the shortest routes on graphs. The problem of routing a single connection as the shortest path is part of the complexity class $\mathcal{P}$. Nonetheless, when the shortest path has to be found for two or more routes and these routes have to compete for a specific scarcity within the confined space (e.g., capacity constraints), the problem becomes an unsplittable multi-commodity flow problem, which is part of the $\mathcal{NP}$-Hard complexity class [94].

This survey will use the term centralized optimization, which stems from [32]. Zhu uses a spectrum to signify the level of dealing with the complexity of limited resources. On the far left of this spectrum is "Independent planning" and on the far right "Centralized planning" [32]. Independent planning implies that the problem of conflicting pipe objectives is completely disregarded. This is often performed by sequentially finding the shortest path for every pipe without taking a routing order into account. Ideally, APR methods should use fully centralized methods, which implies that all limited resources are taken into account such that pipe priorities are globally optimized. Despite the complexity of such fully centralized methods, approximations can be used to take these limited resources into account. Examples of such methods that are implemented in APR literature are often metaheuristic algorithms like: ACO [16, 65, 82], GA [25, 30, 44, 107] and PSO [39]. See Sect. 4.4 for more information on these types of algorithms and how centralized optimization is achieved.

Although sequential routing methods are limited by the problem of competition between pipes, there are multiple techniques that take the global optimum into account while routing pipes sequentially. These methods are: *piping sequence*, *backtracking*, *subset routes* and *macro-pipes*. An overview of these methods is provided below.

Piping Sequence/routing Order The order in which pipes are routed can have a big impact on the final pipe routing design. Jansen et al. [108] show that ordering the sequence of routes based on path length and Minkowski distance can improve routability by over 70%. Therefore, the routing order is often optimized for routability in APR literature. The most commonly used method is to route pipes with the biggest diameters first, see, for example, [6, 8, 37, 69]. It can also be chosen to route short pipes before long pipes [12, 37, 41, 63] and pipes in dense areas can be routed before pipes in sparse areas [12, 41, 63]. Another way to optimize piping routability is to prioritize pipes with the largest surface [6] or biggest volume [81]. A more elaborate method to improve routability is to base the routing order on estimated pipe interferences, such that pipes with a low interference degree can be routed before pipes with a high interference degree [37, 40]. Besides routability, some methods use piping sequences to take material and construction costs into account. One method aims to minimize the route length of expensive pipes by routing them before inexpensive pipes [41, 109]. Thick-walled pipes can also be routed before thin-walled pipes to reduce the number of bends of thick-walled pipes, and subsequently the total bending costs [12, 40, 63]. Accessibility can also be improved with the use of routing order. In aero-engine design, it is advantageous to route pipes that should lay close to the engine before outer pipes [12, 63]. Closely related to this is to route pipes based on the frequency of maintenance, i.e., route pipes with infrequent maintenance before pipes with frequent maintenance [40, 41]. Although there are many objectives that can be improved by the routing order, some methods incorporate the routing order in the algorithm to directly optimize the objective function. Asmara [1] uses discrete particle swarm optimization to iteratively improved the routing order of pipes. Singh and Cheng [78] use simulated annealing to optimize the routing order. Qu et al. [41] uses standard routing sequencing methods which can be adjusted within an improved ant colony optimization algorithm if there is a tie between two pipes.

Backtracking/re-routing Another method to deal with limited resources in the routing space is deleting or re-routing certain pipes as to make place for new ones. Backtracking generally engages itself with choosing which routes to change and how they should be changed [32]. Zhu and Latombe [33] introduce a sophisticated backtracker that chooses pipes based on their relative position and fully reroutes these pipes in a different order. Asmara [1] combines this method with particle swarm optimization to create the hybrid backtracker. Ikehira and Kimura [84] route all pipes separately without collision constraints but re-routes pipes based on the number of collisions such that a feasible solution is found.

Subset Routes The general idea of subset routes is that a finite set of routes is optimized in a global way. One example is provided by Yamada and Teraoka [52], who use the property of multiple shortest paths in Manhattan routing to create a set of feasible shortest paths and second-shortest paths which can be subsequently optimized. Kim and Corne [110] limit routes to basic forms and chooses between these basic forms using stochastic hillclimbing, simulated annealing and a GA. A similar method is used by Ma et al. [20], who use a hierarchical GA that first finds several

good routes, after which the best combination of routes is chosen from this subset of good routes.

Macro-pipes Zhu and Latombe [33] introduce the idea that there are a lot of independent groups of pipes and that these groups can be routed together as macro-pipes. Even though they do not implement this method themselves, it has been used by more recent papers for routing in a space with limited resources. Both Zhao et al. [13] and Dong and Bian [8] make assumptions that certain sets of pipe bundles are provided to attempt to route these pipes as parallel as possible. Yuan et al. [37] uses a clustering method to retrieve sets of pipes that should be routed as macro-pipe.

4.3 Space Dimensionality

Space dimensionality often creates a barrier to the optimization of the model as a result of a combination of two factors. One factor is that the complexity of a pipe routing problem strongly depends on the number of pipes involved. This is because the number of ways to route multiple pipes in the same confined space can get large very quickly when the number of pipes increases. For example, if two pipes can be routed in three different ways, there are eight (2^3) options for a pipe design. If 10 pipes can be routed in three different ways, there are 59,049 (3^{10}) options. As Asmara [1] notes: "Also the complexity of the problem rises exponentially with the number of pipes to be routed." The second factor is the complexity of space. Although the number of pipes to be routed cannot be changed, the number of ways in which these pipes can be routed is highly dependent on how space and routes are modeled. As is mentioned in Sect. 3.1, mathematical models of APR problems are often discretizations of the actual problem. The way the space is modeled is not only important for the description of the problem, but also for its complexity. Take, for example, the grid method used in the previous section as shown in Fig. 4. For this method, the complexity of the problem naturally depends on the number of vertices and edges (grid cells) in the resulting graph. Thus, for combinatorial purposes, it is advantageous to use as few vertices and edges as possible. Nonetheless, using fewer vertices also makes the graph representation of the space less precise, as there are fewer points in space that can be reached. This trade-off between space representation and time complexity is something that should be taken into account when constructing an APR method. Some methods in APR literature try to optimize both space representation and time complexity at the same time. Such methods will be referred to as hierarchical routing and will be discussed in this section.

Hierarchical routing is the idea of using multiple modeling techniques that differ in complexity to infer information and optimize the pipe routing problem in a cooperative way. The general application for hierarchical routing is to first optimize the high-abstraction descriptions with lower complexity, then translate these to more detailed descriptions which can then be used for further optimization and specification. Global and detailed routing are ways in which to describe how specific these descriptions are. Detailed routing refers to the optimization of an APR problem where the location of a pipe is exactly specified. Global routing refers to

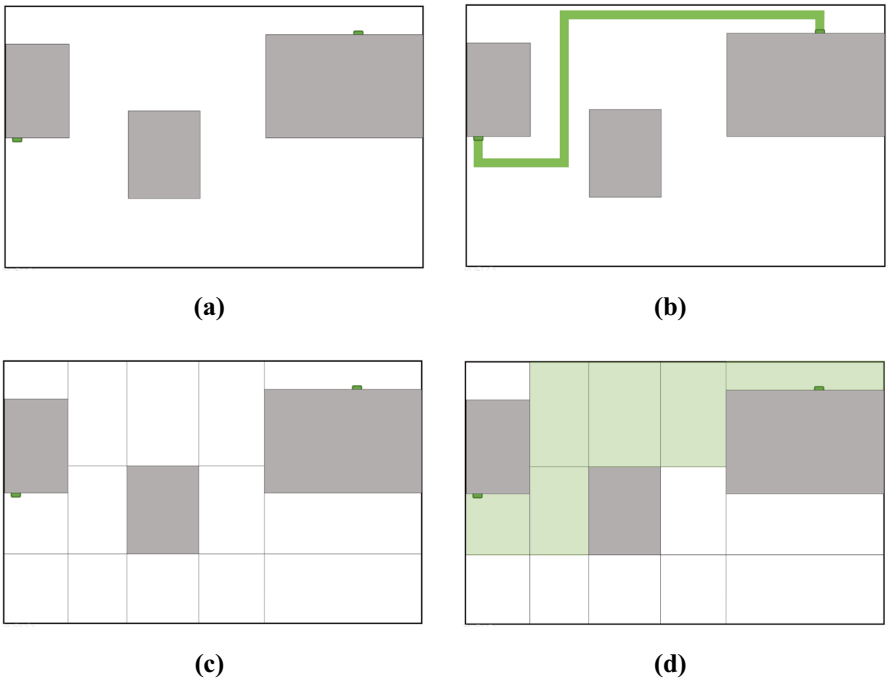


Fig. 13 An example in which the same pipe is routed using detailed routing (**a**) and (**b**) and global routing (**c**) and (**d**)

the optimization of a pipe routing problem where the location of a pipe is not exactly specified. To show the difference between the two cases, an example is provided in Fig. 13.

In Fig. 13a, a confined space is shown with three obstacles in gray and two pipe nozzles in green. These pipe nozzles are the start and end of the pipe to be routed, they also give an indication of the diameter of the pipe. In Fig. 13b, a detailed route is shown for this particular problem, where the exact location of the pipe route is known. In Fig. 13c, the free space is divided into multiple cells using a cell decomposition method. Using this cell decomposition, the global route in Fig. 13d can be found by routing over the cells. This is a global route as it does not indicate where exactly the pipe should be routed, it is only known over which cells to route. An example of hierarchical routing can now be given: first find a global route using the cell decomposition after which a detailed route can be found that goes through the chosen cells of the global route. This particular global cell routing method is applied extensively in the field of Very Large Scale Integration (VLSI) [111].

The method shown in Fig. 13 is not the only method for hierarchical routing. Several categories are made to divide approaches to counter space dimensionality in APR literature. These categories are named: *bounded area search*, *cell hierarchy method*, *point hierarchy method* and *other methods*. An overview of these methods is provided below.

Bounded Area Search Multiple research papers in APR literature use a particular method that this survey will refer to as bounded area search. This method restricts the routing space for every possible route to be within the minimum bounding box of the start and endpoint(s). These minimum bounding boxes can be increased by either a simple constant [41, 69] or they can be increased based on the routing space itself. Bai and Zhang [85], for example, extend these minimum bounding boxes such that blocking obstacles are eliminated. The bounded area search method can also be adapted in an iterative way by using big subspaces that are decreased over time [42]. These methods can be interpreted as hierarchical routing, as the bounded areas can be regarded as global routes.

Cell Hierarchy Method The cell hierarchy method uses cell decomposition to divide a routing space in big cells. Routes are then found by using a number of adjacent cells, after which the adjacent cells are used to find a detailed path. The hierarchical routing method explained above with the help of the illustration in Fig. 13 is an example of a cell hierarchy method. The first research paper that implements global cell routing specifically for APR, uses this exact method [47]. Sometimes only global routing is considered for either cost estimation [77] or flexibility of the resulting design [27]. Zhu [32] uses both a sequential and concurrent method for global cell routing. For the sequential method, global routing is performed by choosing channels, where detailed routes are found within these channels. This process is sequentially repeated for every pipe. For the concurrent method, first, the channels are chosen for every pipe, after which pipes are attempted to be routed through these channels. Asmara [45] uses a cell decomposition method with two levels of hierarchy. Bigger cells are used to first route with the help of Dijkstra's algorithm, then smaller cells are used which are related to the diameters of the pipes. Park and Storch [2] use a cell generation procedure to create adjacent orthogonal cells to create freeways for lower priority pipelines.

Point Hierarchy Method The point hierarchy method is a method to set out a roadmap for detailed routing. The idea of global routing for this method is to use specific points in space that have to be visited. Detailed routing can then be performed by finding specific routes between these points. Figure 14 illustrates how such a method can be implemented. Two points, point 1 (x_1, y_1) and point 2 (x_2, y_2) are set in space. Now a detailed route has to be found that visits both these points. In this way, specific sets of paths can be imposed and other sets of paths can be avoided. Dong and Lin [44] use a GA to place intermediate points after which routes between these points are found using basic styles. Dong and Bian [8] make use of the same method, but implement the A* algorithm to connect these points. Ikehira and Kimura [84] optimize valve placement with the use of a GA. Here, the pipes are routed in detail between the valves using local search algorithms. Kimura [9] extends this method by optimizing equipment placement instead of valve placement, where equipment refers to the set of valves, pumps, T-branches and other connection points. Roh et al. [34] optimize the placement of pipe racks such that they can be used as intermediate points for pipes.

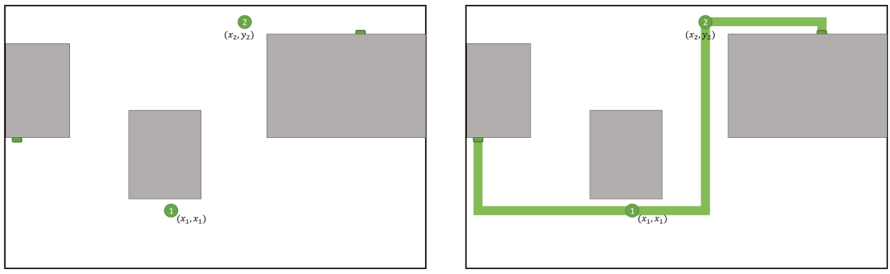


Fig. 14 An example of the point hierarchy method

Other Methods There are other methods in hierarchical pipe routing. Some of these methods are about using the concept of global routing to gain information that is indirectly used in detailed routing, i.e., the global routes do not construct a path that can be used for detailed routing. One example is provided by Yuan et al. [37], where a visibility graph is used to find global optimum routes for pipes to gain information on the interference relation between pipes. Only this information on the interference relation between pipes is used in detailed routing. There are also methods that route pipes while disregarding some constraints. In this way, an approximate route can be found for the pipe which can be adjusted in detailed routing for constraint adherence. One such method is to disregard pipe collisions [67] in global routing. Kang et al. [86] use a 2D sectional plan for global routing while taking a few primary constraints, like hatch covers, main equipment and outfitting, into account. Detailed routing is performed in 3D with an additional set of secondary constraints. Belov et al. [6] routes sample paths without taking into account stress constraints, after which these sample paths can be re-routed in a specific shape to adhere to these constraints.

4.4 Algorithms

Different types of algorithms are used in APR literature. Choosing an algorithm is highly dependent on space, route, objective and constraint modeling. Pathfinding algorithms can only be used when space and routes are represented by a graph, while mathematical optimization depends on the exact modeling of objectives and constraints. How the other optimization barriers are handled also has to be considered in the choice of an algorithm. To have a fully centralized approach (see Sect. 4.2) to multi-pipe routing, it is necessary to choose an algorithm that optimizes all pipes at the same time. As mentioned before, this is often performed with (stochastic) metaheuristic algorithms. An overview of deterministic and stochastic algorithms is provided below accompanied by relevant examples used in APR literature.

4.4.1 Deterministic Algorithms

Deterministic algorithms are algorithms that do not incorporate random sampling. This implies that if a certain set of inputs are used, deterministic algorithms will always provide the same set of outputs. The use of deterministic algorithms in APR literature can be divided into the following two categories: *pathfinding algorithms* and *mathematical optimization*. A short description of both these categories combined with relevant examples is provided below.

Pathfinding Algorithms Pathfinding algorithms are algorithms that explore graphs in such a way that a starting vertex and an ending vertex are connected by a set of adjacent vertices. When pathfinding algorithms are used in APR literature, it is often one of two algorithms that are implemented. One of these algorithms is Dijkstra's algorithm, introduced by Dijkstra [112] (also called lee's algorithm in a grid cell decomposition setting [113]). Dijkstra's algorithm is a simple algorithm that finds the path with the shortest distance between a base vertex and any other vertex in a graph in polynomial time. Note that Dijkstra's algorithm can not be used in graphs with negative weights. Several research papers use Dijkstra's algorithm for pipe routing, see, for example, [55, 58, 105]. Vertex costs for Dijkstra's algorithm are sometimes adjusted to take other factors into account. This can be the number of bends [73, 105], the number of pipe collisions [73] and even the degree of parallel routing in pipes can be included [105]. Ando and Kimura [114] use Dijkstra's algorithm on a grid but allow non-orthogonal bends by smoothing the path fluctuations over multiple grid connections.

The other pathfinding method that is often used in APR literature is called the A* algorithm, introduced by Hart et al. [115]. The A* resembles Dijkstra's algorithm but uses a specific function to determine the cost of a vertex: $f(v) = g(v) + h(v)$. Here, $f(v)$ is the cost of vertex v, $g(v)$ is the current shortest distance to v and $h(v)$ is a heuristic that describes the fitness of v. This fitness value is used to guide the algorithm to a specific target. Therefore, it is often the case that the A* algorithm is a lot faster than Dijkstra's algorithm. Note that if $h(v) = 0\ \forall v$, the A* algorithm collapses into Dijkstra's algorithm. The cost function of the A* algorithm can be interpreted as the costs of the base vertex to the current vertex plus the expected cost from the current vertex to the target. Optimization of pipe routes on a grid is often performed with the A* algorithm, see, for example, [27, 32, 70]. Naturally, the methods that have been implemented for Dijkstra's algorithm can also be implemented for the A* algorithm. Additionally, these methods can also be applied to the heuristic cost component to avoid expected costs in environments that have not yet been explored. These expected costs include the number of bends [27, 32] and the degree of parallel routing [70].

Another interesting pathfinding method is jump point search. Jump point search (JPS) is a method that identifies jump points on a 2D grid such that only a small part of the grid points have to be examined and computation time can be decreased [116]. Min et al. [117] use jump point search for pipe routing optimization in a complex 3D environment. After two experiments, they find that the routing time of JPS was 13,010 times faster (experiment 1) and 388 times faster (experiment 2) than that of A* [117].

Mathematical Programming Most optimization methods discussed in this survey can be regarded as mathematical programming methods, nonetheless, this subsection will be about methods that use systematic approaches for complete declarative mathematical models. One example is given by Shiono et al. [24], who optimize pipe lengths and diameters on a graph using a mixed integer nonlinear program (MINLP). An exact solution was found using an MINLP solver with the help of linear relaxation in pipe diameter constraints. Belov et al. [6] model the pipe routing problem as a mixed integer program that takes several operational, maintenance and installation objectives and constraints into account. They use several state-of-the-art constraint propagation (CP) and mixed integer linear programming (MILP) solvers for optimization. Sakti et al. [81] use constraint programming to pack a free space with cylindrical volumes for pipe routing optimization. Stanczak [62] construct and solve an elaborate MILP model that iteratively uses a different graph for space representation for a continuous approach to piping design.

4.4.2 Stochastic Algorithms

Stochastic algorithms are algorithms that iteratively try to improve on a solution with the use of random sampling. This implies that if a certain set of inputs are used, stochastic algorithms will provide a different set of outputs every time it is used. Stochastic algorithms used in APR literature are often population-based metaheuristics. The biggest advantage of these algorithms is that objectives and constraints do not need to be mathematically tractable, as their inherent structures are only used indirectly by the algorithm. The use of stochastic algorithms in APR literature can be divided into the following categories: *Ant Colony Optimization*, *Particle Swarm Optimization*, *Genetic Algorithms*, *Combinations* and *Other*. A short description of these categories combined with relevant examples is provided below.

Ant Colony Optimization Ant colony optimization (ACO), introduced by Dorigo [118] is a population-based metaheuristic that takes inspiration from the natural behavior of ant colonies [119] to solve discrete optimization problems. Ants communicate with each other by trail pheromones, which are used to mark a path that each ant has walked. Paths that have a higher pheromone intensity are preferred by ants and are thus assigned a higher probability of being chosen. In this way, a stochastic iterative procedure is created that simulates the behavior of ant colonies. Since pheromone intensities are often modeled to dampen over time, short paths will have a higher pheromone intensity than long paths. Pheromone intensities of multiple ants are commonly also modeled to stack on each other, thus the more ants follow a trail the higher pheromone intensity it will have. It can also be chosen to make the pheromone intensity dependent on the length of the path that the ant has walked. In this way, the pheromone intensity of the new-found path can dominate over a longer but more travelled path. Although this is the main idea of ACO, many modifications exist to improve and adapt the algorithm to specific problems. Using ACO for the optimization of APR models is straightforward, as the main objective

of APR is to minimize path length. Fernando [65] gives a broad overview of ACO algorithms that could be used with respect to APR.

As mentioned above, ACO is intended to solve discrete optimization problems. This is the reason that graph methods are often implemented for ACO procedures in APR literature, see, for example, [11, 22, 53, 65]. Thantulage et al. [59] use a standard ACO algorithm based on both a random-based skeleton and grid-based cell decomposition. ACO algorithms can also be adjusted such that they relate more to APR problems, these algorithms are often called Improved ACO (IACO) algorithms. Some examples of such adjustments are to remove unnecessary bends, collisions and pipe lengths (loops) [16, 46, 120]. Qu et al. [41] use an ACO algorithm that is improved with three different kinds of ants that use different moving directions to diversify the searching procedure. Inertia ants prioritize straight lines, general ants search with a variable step size for path diversity and smart ants use a heuristic approach for faster algorithm convergence.

Max-Min Ant Systems (MMAS) are ACO algorithms that have certain bounds for pheromone intensities (max, min) to better control their influence on ants within the system [21]. Another difference with the standard ACO algorithm is that, instead of all ants, only the single best ant updates the pheromone trail to better exploit the best solutions found during the search [121]. Qu et al. [53] use MMAS on both an escape and track graph. Qu et al. [11] uses an MMAS for branched single pipe routing optimization in an octree environment. This method improves search abilities by using a selection of suitable neighbors found using pre-processing techniques and incorporating a dynamic updating mechanism. For better global solutions, multiple ant colonies can be used to represent different pipe branches or even different pipes. Ivić et al. [22] use a cooperative random walk method on a graph to optimize branched pipe routing. Fernando [65] uses ACO for single, multi-objective and multi-pipe routing. Two multi-colony ant systems are used for a centralized approach to multi-pipe routing. One of these algorithms incorporates colony-dependent pheromones such that ants can infer different information from different colonies. Jiang et al. [82] include direction information in the pheromones as a heuristic and use an extension process that increases the pheromone track dependent on the pheromone intensity. This is done to counter premature and slow convergence. Additionally, they implement a centralized optimization approach with the use of co-evolutionary improved multi-ACO (CIMACO). Wu et al. [16] use a dynamic method for pheromone intensity and track selection to avoid premature convergence in local optima.

Particle Swarm Optimization Particle swarm optimization (PSO), introduced by Kennedy and Eberhart [122], is a population-based metaheuristic that takes inspiration from the movement of birds in bird flocks and fish in fish schools [122]. Particles in a swarm iteratively move through the solution space with a specific direction and velocity on the basis of three different components: their current velocity, the position of their personal best solution (cognitive component) and the position of the swarm's best solution (social component). These three components are taken into a single equation to calculate the velocity that determines the location in the next iteration. An example of such a set of equations is:

$$v_{t+1} = \omega v_t + c_1(pb_t - x_t) + c_2(gb_t - x_t), \tag{1}$$

$$x_{t+1} = x_t + v_{t+1} \tag{2}$$

where v_t is the velocity at time t, x_t the position at time t, pb_t the best personal position at time t and gb_t the best global position at time t. Every component has a coefficient that determines its influence. The coefficient that determines the persistence of the current velocity is called the inertia weight (ω), the other two coefficients (c_1, c_2), determining the influence of personal best and global best locations, are called acceleration coefficients [10, 54] or learning factors [28]. Inertia weights can be made dependent on time, such as to prevent premature convergence in local optima [123]. The acceleration coefficients are often multiplied by a random number as to make the iterative procedure of PSO stochastic.

Since many approaches in APR literature use a graph representation, it is also the case that most PSO applications are done on a graph for APR. Most research on PSO methods on graphs in APR literature is performed by Liu and Wang. The authors first use PSO in a grid cell decomposition for a single pipe with variable inertia weights [5]. Two years later, they use PSO to route branched pipelines with both dynamic inertia weights and dynamic acceleration coefficients to control local and global exploration of the search method [124]. The same authors implement PSO with the use of a Discrete PSO (DPSO) algorithm on an escape graph for branched pipelines [10] and make use of a geodesic adjusted visibility graph [54]. The use of PSO for APR on graphs is performed by more researchers. Feng et al. [28] use the chaos grid pre-processing graph and a PSO method modified with an approach that mimics the migration characters of people, where some particles move from underdeveloped locations to more developed places and some brave particles move from developed places to undeveloped places to search for better living spaces. Dong and Lin [39] use multiple swarms with co-evolution on a grid cell decomposition graph to route multiple pipes in a centralized fashion. This is performed with the use of dynamic inertia weights and stochastic acceleration coefficients.

Contrary to ACO, some PSO methods are used in APR literature for continuous routing. Wang et al. [26], who use randomly generated key points where the number of key points depends on the complexity of the space environment and the distance between start and end. These key points are connected using basic styles with the help of insertion points such as to impose orthogonality. Zhao et al. [13] use multi-objective PSO (MOPSO) for continuous multi-pipe routing while minimizing length, pipe smoothness and maximizing parallel pipe routing. Different sets of comparable solutions are provided by clustering the non-dominated sets of the multi-objective results.

Genetic Algorithms The Genetic Algorithm (GA), introduced by Holland [125], is a population-based metaheuristic that mimics the process of natural selection. Every entity in the population has a specific encoding called a chromosome, having several genes to mimic DNA sequences. As with ACO and PSO, iterations (also called

generations for GA) are used to adjust the population over time. Every specific entity has a fitness value directly related to its chromosome. A population and its entities change on the basis of three steps: selection, crossover and mutation. Selection is the way in which two entities are chosen as parents, who pass their genes to a new generation. Entities with a higher fitness value often have a higher probability of becoming parents than entities with a lower fitness value. Crossover is the way in which the genes of the parents are used to construct a new genome for the offspring. Mutation is a technique to randomly change genomes with a small probability. This is often performed on one or a few genes within a genome of an entity.

GAs are frequently used for automated optimization of piping design, see, for example, [29, 44, 80, 107]. Important for GA in APR literature is the way in which pipe routes are encoded. A straightforward method is to find an initial solution for the APR problem, which is then encoded to pipe routes such that it can be adjusted every iteration. One example of this is to relate genes to bend locations as (shown in Fig. 8). Initial routes can be found by simply connecting start and ending points with straight lines, without taking into account any objectives or constraints [42]. Other methods, like Dijkstra's algorithm [71, 72, 80] and an MST [79] are also used for initial solutions. Pipe routes can also be encoded in a constructive way, where an initial solution is not provided. For such a method, in every iteration, the pipe route is extended from the previous point to search for the target [107]. The difference between fixed and variable-length chromosome encoding is also important. Fixed-length encoding means that the number of genes within a chromosome is constant over time, while a variable-length encoding implies it can change over time. Although variable-length encodings increase algorithm complexity, they offer more freedom to express pipe routes [126].

Several modifications can be used to improve the GA for APR purposes. Ito [126] uses a genetic algorithm with the use of two concepts: an eye mechanism that looks ahead to prevent loops and collisions, and a zone concept that uses different priority vectors for different zones. Furuholmen et al. [107] use a coevolutionary GA to design routes for multiple pipes in a constructive way, where every pipe is represented by a full population. Cooperative coevolution is also used by Dong and Lin [44] to route multiple pipes using basic styles between intermediate points. GAs can also be used for continuous pipe routing. Sandurkar and Chen [29] use a genetic encoding based on pipe directions, lengths and bends for the optimization of continuous pipe routing. Other methods use equipment placement encodings [9, 84] and bend location encodings [30] to signify the placement of pipes.

Combinations Combinations between algorithms are often made to exploit multiple patterns at once. One example is to use an A* algorithm for routing pipes but to optimize the routing order using DPSO [1, 45]. Asmara [1] uses a fast line-searching algorithm called Mikami-Tabuchi algorithm to check if a solution exists before routing is performed. Jiang et al. [64] use a multi-colony ACO algorithm that is enhanced with several genetic operators, making it a Multi-ACO-GA algorithm. Kumar and Cheng [15] use a combination of simulated annealing, Dijkstra, 3DA* and fruit fly optimization for combined optimization of both design and installation schedules. Dong and Bian [8] use an improved A*-GA for multi-pipe routing, where

points placed by the genetic algorithm are connected by an improved A* algorithm that takes path length, number of bends and routing along supports into account.

Other Some examples of other nondeterministic algorithms that are used are: Szykman and Cagan [36] use simulated annealing to choose between four options based on a random distribution. These four options are: adding a bend, removing a bend, relocating a bend to a random place and relocating a bend to another place on the same route. Shin et al. [61] use a deep reinforcement learning technique called proximal policy optimization to guide multiple pipes through a space using discrete steps. Ivorra [50] uses the Laminar Navier-Stokes equations to equate pathfinding problems with the natural motion of viscous fluid substances. Liu and Liu [66] use an advanced multi-objective evolutionary algorithm [127] to optimize 3D pipe routing with branches on an improved escape graph.

5 Concise Overview of APR Literature

This section provides an all-encompassing overview of APR literature on the basis of multiple APR research papers that are referenced throughout this survey. The final result is the synthesis table that is split into two tables as shown in Tables 2 and 3. This synthesis table summarizes APR research papers by providing information on the general features of their methodology and by providing information on procedures regarding barriers of both the optimization model and solution algorithm. Sections 3 and 4 have introduced several categories of procedures to deal with these barriers. These categories can consequently be used to give a concise description of how APR research papers deal with such barriers. The features and associated abbreviations within the table are explained below the table.

Some feature entries are denoted by "–" which means that these features are not present in the associated research paper.

- General features
  - *Domain:* This is about the domain for which APR research is performed. Domains can be: Pipe routing in general (*General*), Ship design (*Ship Des.*), Plant layout design (*Plant Des.*), Gas and water distributing systems (*Distr. Sys.*), Aeroengine design (*Aero Eng.*), Cable Routing (*Cable*) and Other (*Other*).
  - *2/3D:* This is about the use of 2D or 3D in the methodology. Note that 3D implies that 3D is used, but 2D could also have been used.
  - *Multi:* This indicates if the research paper routes multiple pipes or only a single one.
  - *Case study:* This is about the size of the case study that is tested. Values range from *Small* (single, non-branched pipes) to *Big* (many-branched multi-pipes within a complex environment). Valuation is based on the number of pipes, the number of obstacles, the complexity of obstacles and the use of branches.

Table 2 An overview of APR literature compiled by different kinds of categories for: 1. General features of the methodology. 2. The procedures for dealing with barriers related to the optimization model. 3. The procedures for dealing with barriers related to the solution algorithm

					Optimization model					
	General features				Space and Route			Constraint modeling		
Source	*Domain*	*2/3D*	*Multi*	*Case study*	*Space*	*Length*	*Bend*	*Physical*	*Safety*	*Operational*
[33]	General	3D	Yes	Medium	CD	Disc.	Disc.	Medium	Medium	Medium
[32]	General	3D	Yes	Big	CD	Disc.	Disc.	High	Low	Low
[45]	Ship Des.	3D	Yes	Medium	CD	Disc.	Disc.	Medium	Low	Low
[1]	Ship Des.	3D	Yes	Big	CD	Disc.	Disc.	High	Medium	Medium
[105]	Ship Des.	3D	Yes	Medium	SK	Disc.	Disc.	Medium	Low	Low
[82]	Ship Des.	3D	Yes	Medium	SK	Disc.	Disc.	Medium	Low	Low
[76]	Distr. Sys.	-	No	Medium	AG	Disc.	Disc.	Medium	Low	Low
[86]	Ship Des.	3D	Yes	Medium	-	Disc.	Disc.	Medium	Low	Low
[10]	Aero Eng.	2D	No	Small	SK	Disc.	Disc.	Medium	Medium	Low
[24]	Distr. Sys.	-	No	Medium	AG	Disc.	Disc.	Low	Low	High
[43]	Plant Des.	2D	No	Small	CD	Disc.	Disc.	Medium	Low	Low
[20]	Cable	-	Yes	Medium	AG	Disc.	Disc.	Low	Low	Low
[18]	Cable	3D	No	Small	AG	Disc.	Disc.	Medium	Low	Low
[55]	Aero Eng.	3D	Yes	Big	AG	Disc.	Disc.	High	Low	Medium
[46]	Ship Des.	3D	No	Small	CD	Disc.	Disc.	Medium	Medium	Medium
[25]	Other	3D	Yes	-	CD	Disc.	Disc.	Medium	Medium	Low
[26]	Other	3D	No	Small	CD + FS	Cont.	Disc.	Medium	Low	Low
[5]	General	2D	No	Small	CD	Disc.	Disc.	Medium	Medium	Low
[11]	Aero Eng.	3D	No	Big	CD	Disc.	Disc.	Medium	Medium	Medium
[42]	Aero Eng.	2D	Yes	Small	CD	Disc.	Disc.	Medium	Low	Low
[63]	Aero Eng.	3D	Yes	Big	FS	Cont.	Cont.	High	Low	High

Table 2 (continued)

					Optimization model					
	General features				Space and Route			Constraint modeling		
Source	*Domain*	*2/3D*	*Multi*	*Case study*	*Space*	*Length*	*Bend*	*Physical*	*Safety*	*Operational*
[39]	Ship Des.	3D	Yes	Medium	SK	Disc.	Disc.	Medium	Low	Low
[12]	Aero Eng.	3D	Yes	Big	FS	Cont.	Cont.	High	Low	High
[27]	Other	3D	No	Medium	CD	Cont.	Disc.	Medium	Low	Medium
[58]	Aero Eng.	3D	Yes	Medium	SK	Disc.	Disc.	Medium	Low	Low
[71]	Ship Des.	3D	Yes	Small	CD	Disc.	Disc.	Medium	Medium	Medium
[64]	Ship Des.	3D	Yes	Small	FS	Cont.	Cont.	Medium	Low	Low
[77]	Plant Des.	3D	Yes	Medium	CD + SK	Disc.	Disc.	Medium	Low	Low
[79]	Plant Des.	2D	Yes	Medium	AG	Disc.	Disc.	Low	High	High
[70]	Aero Eng.	3D	Yes	Small	CD	Disc.	Disc.	Medium	Medium	Low
[52]	Plant Des.	3D	Yes	Small	SK	Disc.	Disc.	Medium	Low	Low
[6]	Plant Des.	3D	Yes	Big	FS	Disc.	Disc.	Medium	Medium	High
[65]	General	3D	Yes	Medium	CD + SK	Both	Both	High	Low	Low
[21]	Distr. Sys.	-	-	Medium	AG	Disc.	Disc.	Low	Low	High
[50]	General	3D	No	Small	FS	Cont.	Cont.	High	Low	Low
[83]	Ship Des.	3D	Yes	Small	CD	Disc.	Disc.	Medium	Medium	Medium
[67]	Ship Des.	3D	Yes	Medium	FS	Disc.	Disc.	Medium	Low	Low
[84]	Ship Des.	3D	Yes	Medium	FS	Disc.	Disc.	Medium	Low	High
[9]	Ship Des.	3D	Yes	Medium	FS	Disc.	Disc.	Medium	Low	High
[114]	Ship Des.	3D	No	Small	SK	Disc.	Disc.	Medium	Low	Low
[78]	Plant Des.	3D	Yes	Medium	SK	Disc.	Disc.	Medium	Low	Medium
[15]	Plant Des.	3D	Yes	Medium	SK	Disc.	Disc.	High	High	High
[53]	Aero Eng.	3D	Yes	Medium	AG	Disc.	Disc.	Medium	Medium	Low

Table 2 (continued)

	Optimization model						**Solution algorithm**			
	Constraint modeling	Obstacle modeling								
Source	*Maintenance*	*Material*	*Construction*	*Operational*	*Maintenance*	*Aesthetics*	*Branch*	*Space dim.*	*Lim. resources*	*Algorithm*
[33]	Medium	Medium	Medium	Medium	Medium	Low	Seq.	CHM	Order + BT	A*
[32]	Low	Medium	Low	Low	Low	Low	-	CHM	Order + BT	A*
[45]	Low	Medium	Low	Low	Low	Low	Seq.	CHM	Order	Dijkstra + DPSO
[1]	Medium	Medium	Medium	Medium	Medium	Low	Seq.	-	Order + BT	A* + DPSO for order
[105]	Low	Medium	Medium	Low	Low	Low	Seq.	-	-	Adjusted Dijkstra
[82]	Low	Medium	Medium	Low	Low	Low	Seq.	-	MetaH	IACO + CIMACO
[76]	Low	Medium	Low	Low	Low	Low	Par.	-	-	Depth-first search
[86]	Medium	Medium	Low	Low	Low	Medium	-	Other	-	Other
[10]	Low	Medium	Medium	Low	Low	Low	Par.	-	-	Discrete PSO
[24]	Low	Medium	Low	Low	Low	Low	Exact	-	-	MINLP solver + linear relaxation
[43]	Low	Medium	Medium	Low	Low	Low	-	-	-	GA
[20]	Low	Medium	Low	Low	Low	Low	-	Other	SR	Hierarchical GA
[18]	Low	Medium	Low	Low	Low	Low	Par.	-	-	Dijkstra + GA
[55]	Medium	Medium	Low	Medium	Low	Low	-	-	BT	Adjusted Dijkstra
[46]	Medium	Medium	Medium	Low	Low	Low	-	-	-	IACO and HCCIACO
[25]	Low	Medium	Medium	Medium	Low	Low	-	-	MetaH	GA
[26]	Low	Medium	Medium	Low	Medium	Low	-	-	-	Other
[5]	Low	Medium	Medium	Low	Low	Low	-	-	-	Modified PSO
[11]	Medium	Medium	Medium	Medium	Low	Low	Par.	-	-	Modified MMAS
[42]	Medium	Medium	Low	Medium	Medium	Low	-	BAS	-	GA
[63]	Medium	Medium	Low	Low	Medium	Low	-	BAS	Order	Other

Table 2 (continued)

	Optimization model						**Solution algorithm**			
	Constraint modeling	Obstacle modeling								
Source	*Maintenance*	*Material*	*Construction*	*Operational*	*Maintenance*	*Aesthetics*	*Branch*	*Space dim.*	*Lim. resources*	*Algorithm*
[39]	Low	Medium	Medium	Low	Low	Medium	Par.	-	MetaH	Multi-PSO with co-evolution
[12]	Medium	Medium	Medium	Low	Medium	Low	-	BAS	Order	Other
[27]	Low	Medium	Medium	Low	Low	Low	-	CHM (GO)	-	A* + LP
[58]	Low	Medium	Low	Low	Low	Low	-	-	-	Dijkstra
[71]	Medium	Medium	Medium	Low	Low	Low	-	-	MetaH	Dijkstra + GA
[64]	Low	Medium	Medium	Low	Low	Low	Seq.	-	MetaH	Multi-ACO-GA Algorithm
[77]	Low	Medium	Medium	Low	Low	Low	-	CHM (GO)	-	Dijkstra + Vector router
[79]	Low	Medium	Medium	High	Low	Low	MST	(GO)	-	MST + LP
[70]	Low	Medium	Medium	Low	Low	Low	Seq.	-	-	A*
[52]	Low	Medium	Low	Low	Low	Low	-	-	SR	Other
[6]	Medium	Medium	Medium	High	Medium	Low	-	Other	Order + BT	CP + MINLP solvers
[65]	Low	Medium	Medium	Low	Low	Low	-	-	MetaH	PSACO and P-ACO
[21]	Low	Low	Low	High	Low	Low	Par.	-	-	MMAS
[50]	Low	Medium	Low	Low	Low	Low	-	-	-	Laminar Navier-Stokes equations
[83]	Low	Medium	Low	Medium	Low	Low	-	-	Order	A*
[67]	Low	Medium	Low	Medium	Medium	Low	-	Other	BT	MOGA
[84]	Low	Medium	Low	High	Low	Low	Seq.	PHM	BT	Multi-Objective GA (NSGA-II)
[9]	Low	Medium	Low	High	Low	Low	Seq.	PHM	-	Multi-Objective GA (NSGA-II)
[114]	Low	Medium	Medium	Medium	Low	Low	-	-	-	Dijkstra
[78]	Low	Medium	Medium	High	Low	Low	Seq.	-	Order	(3D A*, FOA, Dijkstra) + SA
[15]	Medium	High	High	High	Medium	Low	Seq.	-	Order	(3D A*, FOA, Dijkstra) + SA
[53]	Low	Medium	Medium	Low	Low	Low	Par.	-	-	Concurrent ACO + MMAS

Table 3 An overview of APR literature compiled by different kinds of categories for: 1. General features of the methodology. 2. The procedures for dealing with barriers related to the optimization model. 3. The procedures for dealing with barriers related to the solution algorithm

					Optimization model					
	General features				Space and Route			Constraint modeling		
Source	*Domain*	*2/3D*	*Multi*	*Case study*	*Space*	*Length*	*Bend*	*Physical*	*Safety*	*Operational*
[80]	Ship Des.	3D	No	Small	CD	Disc.	Disc.	Medium	Low	Low
[109]	Plant Des.	3D	Yes	Medium	AG	Disc.	Disc.	Medium	Medium	Low
[81]	General	3D	Yes	Medium	FS	Disc.	Disc.	High	Low	Low
[22]	Distr. Sys.	-	No	Small	AG	Disc.	Disc.	Low	Low	High
[85]	Aero Eng.	3D	Yes	Big	-	Disc.	Disc.	Medium	Low	Medium
[74]	Ship Des.	3D	No	Small	FS	Disc.	Disc.	Medium	Low	Low
[107]	General	3D	Yes	Medium	CD	Disc.	Disc.	Medium	Low	Low
[117]	Ship Des.	3D	No	Small	CD	Disc.	Disc.	Medium	Low	Low
[29]	General	3D	No	Small	FS	Cont.	Both	High	Low	Low
[23]	Distr. Sys.	-	No	Small	AG	Disc.	Disc.	Low	Low	High
[59]	General	3D	No	Small	CD + SK	Both	Both	High	Low	Low
[37]	Aero Eng.	3D	Yes	Medium	SK	Disc.	Disc.	High	Medium	Medium
[128]	Ship Des.	3D	-	-	-	Disc.	Disc.	Medium	Low	Medium
[14]	Plant Des.	2D	No	Small	CD	Disc.	Disc.	Medium	Low	Low
[16]	Plant Des.	3D	Yes	Medium	SK	Disc.	Disc.	Medium	Low	Low
[110]	Plant Des.	2D	Yes	Small	AG	Disc.	Disc.	Medium	Low	Low
[129]	Ship Des.	3D	No	-	-	Disc.	Disc.	Medium	Low	Low
[130]	Ship Des.	-	Yes	-	-	Disc.	Disc.	Low	Low	Low
[66]	General	3D	No	Medium	SK	Disc.	Disc.	Medium	Medium	Low
[13]	Aero Eng.	2D	Yes	Small	-	Cont.	Cont.	Medium	Low	Low
[54]	Aero Eng.	2D	No	Small	AG	Disc.	Disc.	Medium	Low	Low

Table 3 (continued)

					Optimization model					
	General features				Space and Route			Constraint modeling		
Source	*Domain*	*2/3D*	*Multi*	*Case study*	*Space*	*Length*	*Bend*	*Physical*	*Safety*	*Operational*
[108]	General	3D	Yes	Big	AG	Disc.	Disc.	Medium	Low	Low
[62]	General	3D	No	Small	FS	Cont.	Cont.	Low	Low	Low
[41]	Aero Eng.	3D	Yes	Big	CD	Disc.	Disc.	High	Medium	Medium
[75]	Distr. Sys.	2D	No	Medium	AG	Disc.	Disc.	Low	Low	Low
[61]	Ship Des.	3D	Yes	Medium	FS	Disc.	Disc.	Medium	Medium	Medium
[124]	Aero Eng.	2D	Yes	Medium	CD	Disc.	Disc.	Medium	Low	Low
[2]	Ship Des.	3D	Yes	Medium	CD	Disc.	Disc.	High	Medium	High
[28]	Other	3D	Yes	Medium	SK	Disc.	Disc.	Medium	Low	Low
[73]	Ship Des.	2D	Yes	Small	CD	Disc.	Disc.	Medium	Medium	Low
[3]	Plant Des.	3D	Yes	Medium	-	Disc.	Disc.	Low	Low	Low
[40]	Aero Eng.	3D	Yes	Big	FS	Cont.	Cont.	High	Low	Low
[34]	Ship Des.	3D	Yes	Big	-	Cont.	Disc.	High	Low	Low
[72]	Aero Eng.	3D	Yes	Medium	SK	Disc.	Disc.	Medium	Low	Medium
[131]	Aero Eng.	3D	-	-	-	Disc.	Disc.	High	-	-
[126]	General	2D	No	Small	CD	Disc.	Disc.	Medium	Low	Low
[8]	Ship Des.	3D	Yes	Medium	CD	Disc.	Disc.	Medium	Medium	Medium
[69]	Ship Des.	3D	Yes	Medium	CD	Disc.	Disc.	High	Medium	Low
[44]	Ship Des.	3D	Yes	Medium	CD	Disc.	Disc.	Medium	Medium	Low
[132]	Aero Eng.	3D	-	-	CD	Disc.	Disc.	High	-	-
[36]	Plant Des.	3D	Yes	Medium	FS	Cont.	Cont.	Low	Low	Low
[120]	Ship Des.	3D	No	Small	AG	Disc.	Disc.	Medium	Medium	Low
[31]	Ship Des.	3D	No	Small	CD + SK	Disc.	Disc.	High	Low	Low

Table 3 (continued)

					Optimization model					
	General features				Space and Route			Constraint modeling		
Source	*Domain*	*2/3D*	*Multi*	*Case study*	*Space*	*Length*	*Bend*	*Physical*	*Safety*	*Operational*
[47]	Ship Des.	3D	Yes	Small	CD	Disc.	Disc.	Medium	Low	Medium
[30]	Other	3D	Yes	Small	FS	Cont.	Cont.	Medium	Low	Low

	Optimization model						**Solution algorithm**			
	Constraint modeling	Obstacle modeling								
Source	*Maintenance*	*Material*	*Construction*	*Operational*	*Maintenance*	*Aesthetics*	*Branch*	*Space dim.*	*Lim. resources*	*Algorithm*
[80]	Low	Medium	Low	Low	Low	Low	Seq.	-	-	Dijkstra + GA
[109]	Low	Medium	Medium	Low	Low	Medium	-	-	Order	Dijkstra + Vector router
[81]	Low	Medium	Low	Low	Low	Low	-	-	Order	MILP + search heuristic
[22]	Low	Medium	Low	Medium	Low	Low	Par.	-	-	Cooperative Random Walk (ACO)
[85]	Low	Medium	Medium	Medium	Medium	Medium	-	BAS	-	Other
[74]	Medium	Medium	Low	Low	Low	Low	-	-	-	Other
[107]	Low	Medium	Low	Low	Low	Low	-	-	MetaH	Standard + incremental + evolut. GA
[117]	Low	Medium	Medium	Low	Low	Low	-	-	-	Adjusted JPS
[29]	Low	Medium	Low	Low	Low	Low	-	-	-	GA
[23]	Low	Medium	Low	Medium	Medium	Low	Par.	-	-	Dynamic programming + GA
[59]	Low	Medium	Low	Low	Low	Low	-	-	-	ACO
[37]	Low	Medium	High	High	High	Low	-	Other	Order + MP	A* + EA
[128]	Medium	Low	Low	Low	Low	Low	-	-	-	Other
[14]	Low	Medium	Medium	Low	Low	Low	-	-	-	GA + Rule based inference

Table 3 (continued)

	Optimization model						**Solution algorithm**			
	Constraint modeling	Obstacle modeling								
Source	*Maintenance*	*Material*	*Construction*	*Operational*	*Maintenance*	*Aesthetics*	*Branch*	*Space dim.*	*Lim. resources*	*Algorithm*
[16]	Medium	Medium	Medium	Low	Low	Low	-	-	MetaH	Modified ACO
[110]	Low	Medium	Low	Low	Low	Low	-	-	SR	Stochastic hillclimbing + SA + GA
[129]	Low	Low	Low	Low	Low	Low	-	-	-	Other
[130]	Medium	Medium	Low	Low	Low	Low	-	(GO)	-	Other
[66]	Low	Medium	Medium	Low	Low	Low	MST	-	-	Multi Objective EA
[13]	Low	Medium	Medium	Low	Low	Medium	-	-	MP	Multi Objective PSO
[54]	Low	Medium	Low	Low	Low	Low	MST	-	-	MST + PSO
[108]	Low	Low	Low	Low	Low	Low	-	-	Order	A*
[62]	Low	Medium	Medium	Low	Low	Low	-	-	-	MILP solver
[41]	High	Medium	High	Medium	High	High	-	BAS	Order	Improved ACO
[75]	Low	Medium	Low	Low	Low	Low	Par.	-	-	ACO
[61]	Medium	Medium	Medium	Low	Low	Low	-	-	Order	Reinforcement Learning (PPO)
[124]	Low	Medium	Medium	Medium	Low	Medium	-	-	Order	DPSO
[2]	Low	High	High	High	Low	Low	Multiple	CHM	Order	Other
[28]	Low	Medium	Medium	Low	Low	Low	Seq.	-	MetaH	Modified PSO
[73]	Medium	Medium	Low	Low	Medium	Low	-	-	-	Adjusted Dijkstra
[3]	Low	Medium	Low	Low	Low	Low	Seq.	-	-	Other
[40]	Low	Medium	Low	Low	Medium	Low	-	-	Order	Other
[34]	Low	Medium	High	Low	Low	Low	-	PHM	MP	Other
[72]	Low	Medium	Low	Low	Low	Low	-	-	-	Dijkstra + GA
[131]	-	-	-	-	-		-	-	-	Other

Table 3 (continued)

	Optimization model						**Solution algorithm**			
	Constraint modeling	Obstacle modeling								
Source	*Maintenance*	*Material*	*Construction*	*Operational*	*Maintenance*	*Aesthetics*	*Branch*	*Space dim.*	*Lim. resources*	*Algorithm*
[126]	Low	Medium	Medium	Low	Low	Low	-	-	-	Improved GA
[8]	Low	Medium	High	Medium	Medium	Medium	Seq.	PHM	Order + MP	GA + Improved A*
[69]	Medium	Medium	Medium	Low	Low	Low	Seq.	BAS	Order	Dijkstra + NSGA-II + CCNSGA-II
[44]	Medium	Medium	Medium	Medium	Medium	Medium	Par.	PHM	MetaH	GA + CCGA
[132]	-	-	-	-	-	-	-	-	-	Other
[36]	Low	Medium	Low	Medium	Low	Low	-	-	-	SA
[120]	Low	Medium	Medium	Medium	Low	Low	-	-	-	Improved ACO
[31]	Low	Medium	Medium	Medium	Low	Low	-	-	-	Adjusted Dijkstra
[47]	Low	High	Low	Low	Low	Low	-	CHM (GO)	BT	Dynamic programming
[30]	Low	Medium	Low	Low	Low	Low	-	-	MetaH	GA

- Optimization model (Sect. 3)
  - *Space:* This is about the space representation that is used. Space modeling can be done by: Cell decomposition (*CD*), Skeletonization (*SK*), Assumed graph (*AG*), Free space (*FS*), or any combination of these.
  - *Length:* This is about the length-continuity of pipes. The values can be: Discrete (*Disc*), Continuous (*Cont.*), or Discrete and Continuous (*Both*).
  - *Bend:* This is about the bend continuity of pipes. The values can be: Discrete (*Disc*), Continuous (*Cont.*), or Discrete and Continuous (*Both*).
  - *Constraint modeling:* For every type of constraint, a valuation is provided that signifies to what extent this type of constraint is handled in the research paper. Values range from *Low* to *High*. Valuation is based on the number and complexity of implemented constraints.
  - *Objective modeling:* For every type of objective, a valuation is provided that signifies to what extent this type of objective is handled in the research paper. Values range from *Low* to *High*. Valuation is based on the number and complexity of implemented objectives.
- Solution algorithm (Sect. 4)
  - *Branch:* This is about the procedures that are used to deal with the branching barrier. Methods that can be used are: Sequential branching (*Seq.*), Minimum spanning trees (*MST*), Parallel branching (*Par.*), Exact Branching (*Exact*) or any combination of these (*Multiple*).
  - *Space dim.:* This is about the procedures that are used to deal with the space dimensionality barrier. Methods that can be used are: Bounded area search (*BAS*), Cell hierarchy method (*CHM*), Point hierarchy method (*PHM*) and Other methods (*Other*). Sometimes *(GO)* is used to indicate that global routing is performed, but detailed routing is not performed.
  - *Lim. resources:* This is about the procedures that are used to deal with the resource competition barrier. Methods that can be used are: Piping order (*Order*), Backtracking/re-routing (*BT*), Subset routes (*SR*), Macro-pipes (*MP*) or Fully centralized optimization with the use of metaheuristics (*MetaH*).
  - *Algorithm:* This is about the algorithm or combinations of algorithms that are used.

Note that this section is not intended to rank the methods provided in the given sources. It is solely about the extent to which the content of these papers can assist readers in reproducing such methods. For example, if a paper mentions certain constraints, but does not show if or how they are implemented, then it does not prove useful for the reader and will not be taken into account in the synthesis table.

6 Discussion

This survey provides an overview of APR literature and the methods they have proposed. The three main goals of this survey have been addressed as follows: Section 2 offers a basic description of APR problems. Sections 3 and 4 explain and categorize

the most important barriers that obstruct the implementation of APR methods (goals 1 and 2). Section 5 then provides a synthesis table that gives a concise overview of APR literature (goal 3).

Although the possibility of an adequate APR method becomes more and more plausible, contemporary approaches still seem to be inadequate to be used without supervision or expert input. Thus, for practical purposes, it is important for future methods that user interaction is increased. This is useful for assisting piping engineers to complete the piping design at an expert level by reducing human errors [86], but also to train new piping engineers [133]. Finally, it could also prove useful to ease the transition to a fully functional APR system in the future. One such method could be to improve the interaction between CAD models and the APR optimizer to assist piping engineers with visual results [1, 31, 134].

The limitation of APR literature is that although there are some important variables that may have an indirect influence on pipe routing optimization, they are not taken into account. One example is layout optimization, which is about the placement of equipment and/or machinery. Layout optimization is very important for pipe routing because most pipes have to be connected to equipment, which means that an APR problem is highly dependent on the layout. Even though joint optimization would be preferred [21], the application of this method is limited in the literature. A few examples can be found in [81, 109].

There are many future research possibilities on the subject of APR. One of these unexplored territories is the introduction of uncertainty into pipe routing. In practice, uncertainty is common in piping design. Uncertainty can happen in several places. They can happen in how the space looks like, i.e., the confined space boundaries and obstacles. They can also happen in pipes, i.e., which pipes, their diameters, or their starting and ending positions. Uncertainty can also happen in objectives and constraints, think about the fluctuating price of materials. One potent example is the transition of the maritime industry towards zero-carbon energy sources. Because the future of this energy transition is very uncertain and ships are built for long time periods, it is expected that ships that are built right now have to transition somewhere in their lifetime. Taking this uncertainty into account can decrease future transition costs. There are a few examples of APR methods that take a small amount of uncertainty into account. One such method has been developed in the domain of APR in ship design by Roh et al. [34]. They propose a method that links pipe routes to the hull structure of the ship. This method is developed to construct a piping design that is robust for future structural changes in the ship's hull. Asmara [1] uses sensitivity analysis to check the robustness of the piping design. Since research on uncertainty in graph theory is abundant, a connection between the two fields could prove to be useful in the future.

Funding This publication is part of the project READINESS with project number TWM.BL.019.002 of the research program *"Topsector Water & Maritime: the Blue route"* which is partly financed by the Dutch Research Council (NWO).

Data Availability This paper has no associated data.

Declarations

Conflict of Interest The author declares no competing interests.

References

1. Asmara A (2013) Pipe routing framework for detailed ship design. PhD thesis, Delft University of Technology. https://repository.tudelft.nl/islandora/object/uuid%3A4da99646-37d7-49e5-9d3b-51de81ba29dd
2. Park JH, Storch RL (2002) Pipe-routing algorithm development: case study of a ship engine room design. Expert Syst Appl 23(3):299–309. https://doi.org/10.1016/S0957-4174(02)00049-0
3. Calixto E, Bordeira P, Calazans H et al (2009) Plant design project automation using an automatic pipe routing routine. Comput Aided Chem Eng 27:807–812. https://doi.org/10.1016/S1570-7946(09)70355-4
4. Moran S (ed) (2017) Process plant layout, 2nd edn. Butterworth-Heinemann, Oxford, https://doi.org/10.1016/B978-0-12-803355-5.00040-8
5. Liu Q, Wang C (2008) A modified particle swarm optimizer for pipe route design. In: 2008 11th IEEE International Conference on Computational Science and Engineering - Workshops, pp 157–161. https://doi.org/10.1109/CSEW.2008.29
6. Belov G, Czauderna T, Dzaferovic A et al (2017) An optimization model for 3D pipe routing with flexibility constraints. In: Beck JC (ed) Principles and Practice of Constraint Programming. Springer International Publishing, Cham, pp 321–337. https://doi.org/10.1007/978-3-319-66158-2_21
7. Qian X, Ren T, Wang C (2008) A survey of pipe routing design. In: 2008 Chinese Control and Decision Conference. pp 3994–3998. https://doi.org/10.1109/CCDC.2008.4598081
8. Dong Z, Bian X (2020) Ship pipe route design using improved A* al-gorithm and genetic algorithm. IEEE Access 8:153273–15329. https://doi.org/10.1109/ACCESS.2020.3018145
9. Kimura H (2011). Automatic designing system for piping and instruments arrangement including branches of pipes. International Conference on Computer Applications in Shipbuilding 2011. https://doi.org/10.2534/JJASNAOE.14.165
10. Liu Q, Wang C (2011) A discrete particle swarm optimization algorithm for rectilinear branch pipe routing. Assem Autom 31:363–368. https://doi.org/10.1108/01445151111172952
11. Qu YF, Jiang D, Zhang XL (2018b) A new pipe routing approach for aero-engines by octree modeling and modified max-min ant system optimization algorithm. J Mech 34:1–9. https://doi.org/10.1017/jmech.2016.86
12. Yin Y, Zhou C, Zhu J (2010) A pipe route design methodology by imitating human imaginal thinking. CIRP Ann Manuf Technol 59:167–170. https://doi.org/10.1016/j.cirp.2010.03.096
13. Zhao H, Liu Q, Tong B (2019) Multi-pipe routing in bundles for aero-engine using MOPSO. DEStech Trans Comput Sci Eng 927–933. In: 2019 International Conference on Computation and Information Sciences (ICCIS 2019). https://doi.org/10.12783/dtcse/iccis2019/31995
14. Ito T, Fukuda S (1998) Hybrid approach to piping route path design using GA-based inspiration and rule-based inference. Concurr Eng 6(4):323–332. https://doi.org/10.1177/1063293X9800600405
15. Kumar S, Cheng J (2015) A BIM-based automated site layout planning framework for congested construction sites. Autom Constr 59:24–37. https://doi.org/10.1016/j.autcon.2015.07.008
16. Wu L, Tian X, Wang H et al (2019) Improved ant colony optimization algorithm and its application to solve pipe routing design. Assem Autom 39:45–57. https://doi.org/10.1108/AA-02-2018-022

17. Cazzaro D, Fischetti M, Fischetti M (2020) Heuristic algorithms for the Wind Farm Cable Routing problem. Appl Energy 278(115):617. https://doi.org/10.1016/j.apenergy.2020.115617
18. Conru A (1994) A genetic approach to the cable harness routing problem, vol 1. In: Proceedings of the First IEEE Conference on Evolutionary Computation. IEEE World Congress on Computational Intelligence. pp 200–205. https://doi.org/10.1109/ICEC.1994.350016
19. Duvnjak Zarkovic S, Shayesteh E, Hilber P (2021) Onshore wind farm - reliability centered cable routing. Electr Power Syst Res 196(107):201. https://doi.org/10.1016/j.epsr.2021.107201
20. Ma X, Iida K, Xie M et al (2006) A genetic algorithm for the optimization of cable routing. Syst Comput Japan 37:61–71. https://doi.org/10.1002/scj.10250
21. Afshar MH (2006) Application of a max-min ant system to joint layout and size optimization of pipe networks. Eng Optim 38(3):299–317. https://doi.org/10.1080/03052150500476357
22. Ivić S, Grbčić L, Družeta S (2016) Cooperative random walk for pipe network layout optimization. Int J Appl Eng Res 11:2839–2847. https://www.semanticscholar.org/paper/Cooperative-Random-Walk-for-Pipe-Network-Layout-Ivi%C4%87-Grb%C4%8Di%C4%87/a609f15a240ab8eed2593b647cf87406575ea3a8
23. Savic DA, Walters GA (1995) Genetic operators and constraint handling for pipe network optimization. In: Fogarty TC (ed) Evolutionary Computing. Springer Berlin Heidelberg, Berlin, Heidelberg, pp 154–165. https://doi.org/10.1007/3-540-60469-3_32
24. Shiono N, Suzuki H, Saruwatari Y (2019) A dynamic programming approach for the pipe network layout problem. Eur J Oper Res 277(1):52–61. https://doi.org/10.1016/j.ejor.2019.02.036
25. Wang C, Sun X, Yuan T (2015) A method based genetic algorithm for pipe routing design. In: Proceedings of the 2015 International Conference on Advanced Engineering Materials and Technology. https://doi.org/10.2991/icaemt-15.2015.156
26. Wang C, Sun L, Sun X et al (2016) A method based on PSO for pipe routing design. In: 2016 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER), pp 422–427. https://doi.org/10.1109/CYBER.2016.7574862
27. Stanczak M, Pralet C, Vidal V et al (2021). A pipe routing hybrid approach based on a-star search and linear programming. Lect Notes Comp Sci 12735:179–195. In: International Conference on Integration of Constraint Programming, Artificial Intelligence, and Operations Research (CPAIOR 2021). https://doi.org/10.1007/978-3-030-78230-6_12
28. Feng H, Fu Y, Li R (2007) Pipe-routing algorithm for pipelines with branches. Appl Mech Mater 10–12:430–434. https://doi.org/10.4028/0-87849-470-7.430
29. Sandurkar S, Chen W (2000) GAPRUS - Genetic algorithms based pipe routing using tessellated objects. Comput Ind 38:209–223. https://doi.org/10.1016/S0166-3615(98)00130-4
30. Wang H, Zhao C, Yan W et al (2006) Three-dimensional multi-pipe route optimization based on genetic algorithms. International Federation for Information Processing Digital Library; Knowledge Enterprise: Intelligent Strategies in Product Design, Manufacturing, and Management 207:177–183. https://doi.org/10.1007/0-387-34403-9_23
31. Kim SH, Ruy WS, Jang BS (2013) The development of a practical pipe auto-routing system in a shipbuilding CAD environment using network optimization. Int J Nav Archit Ocean Eng 5(3):468–477. https://doi.org/10.2478/IJNAOE-2013-0146
32. Zhu D (1992) Exploring the interaction of geometry and search in path planning. PhD thesis, Stanford University, California
33. Zhu D, Latombe J (1991) Pipe routing-path planning (with many constraints). In: Proceedings. 1991 IEEE International Conference on Robotics and Automation. IEEE Computer Society, Los Alamitos, CA, USA, pp 1940–1947. https://doi.org/10.1109/ROBOT.1991.131911
34. Roh MI, Lee KY, Choi WY (2007) Rapid generation of the piping model having the relationship with a hull structure in shipbuilding. Adv Eng Softw 38:215–228. https://doi.org/10.1016/j.advengsoft.2006.10.002
35. Bellot F, Krause E (1988) Taxicab geometry: an adventure in non-euclidean geometry. Math Gaz 72:255. https://doi.org/10.2307/3618288
36. Szykman S, Cagan J (1996) Synthesis of optimal nonorthogonal routes. J Mech Des 118(3):419–424. https://doi.org/10.1115/1.2826902
37. Yuan H, Yu J, Jia D et al (2021) Group-based multiple pipe routing method for aero-engine focusing on parallel layout. Front Mech Eng 16:798–813. https://doi.org/10.1007/s11465-021-0645-3
38. Dielissen VJ, Kaldewaij A (1991) Rectangular partition is polynomial in two dimensions but NP-complete in three. Inf Process Lett 38:1–6. https://doi.org/10.1016/0020-0190(91)90207-X
39. Dong ZR, Lin Y (2017) A particle swarm optimization based approach for ship pipe route design. Int Shipbuild Prog 63:59–84. https://doi.org/10.3233/ISP-160123

40. Wang C, Liu Q (2011) Projection and geodesic-based pipe routing algorithm. IEEE Trans Autom Sci Eng 8:641–645. https://doi.org/10.1109/TASE.2010.2099219
41. Qu Y, Jiang D, Gao G et al (2015) Pipe routing approach for aircraft engines based on ant colony optimization. J Aerosp Eng 29(04015):057. https://doi.org/10.1061/(ASCE)AS.1943-5525.0000543
42. Ren T, Zhu ZL, Dimirovski G et al (2013) A new pipe routing method for aero-engines based on genetic algorithm. Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering 228:424–434. https://doi.org/10.1177/0954410012474134
43. Ito T (1999) A genetic algorithm approach to piping route path planning. J Intell Manuf 10(1):103–114. https://doi.org/10.1023/A%3A1008924832167
44. Dong Z, Lin Y (2017) Ship Pipe routing method based on genetic algorithm and cooperative coevolution. J Sh Prod Des 33:122–134. https://doi.org/10.5957/JSPD.33.2.150005
45. Asmara A (2006) Automatic piping system in ship. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.105.354&rep=rep1&type=pdf. Paper presented at the International Conference on Computer and IT Application (COMPIT)
46. Wang Y, Yu Y, Li K et al (2018) A human-computer cooperation improved ant colony optimization for ship pipe route design. Ocean Eng 150:12–20. https://doi.org/10.1016/j.oceaneng.2017.12.024
47. Van der Tak C (1976) The optimum routeing of pipes in a ship's engine room. Paper presented at the Computer Applications in the Automation of Shipyard Operation and Ship Design II, ICCAS '76. https://repository.tudelft.nl/islandora/object/uuid%3A2e1ed503-51d3-4b1b-acaf-b00558abfe4e
48. Mehta D, Blust G (1997) Corner stitching for simple rectilinear shapes [VLSI layouts]. IEEE Trans Comput Aided Des Integr Circuits Syst 16(2):186–198. https://doi.org/10.1109/43.573833
49. Ousterhout J (1984) Corner stitching: a data-structuring technique for VLSI layout tools. IEEE Trans Comput Aided Des Integr Circuits Syst 3(1):87–100. https://doi.org/10.1109/TCAD.1984.1270061
50. Ivorra B (2018) Application of the Laminar Navier-Stokes equations for solving 2D and 3D path-finding problems with static and dynamic spatial constraints: implementation and validation in Comsol Multiphysics. J Sci Comput 74:1163–1187. https://doi.org/10.1007/s10915-017-0489-5
51. Lingas A, Pinter R, Rivest R et al (1982) Minimum edge length partitioning of rectilinear polygons. Proceedings - Annual Allerton Conference on Communication, Control, and Computing, pp 53–63. https://www.scopus.com/record/display.uri?eid=2-s2.0-0020227929 &origin=inward & txGid=d67439ff1a4f65c08b10b58e12ad0c14
52. Yamada Y, Teraoka Y (1998) An optimal design of piping route in a CAD system for power plant. Comput Math with Appl 35(6):137–149. https://doi.org/10.1016/S0898-1221(98)00025-X
53. Qu Y, Jiang D, Yang Q (2018a) Branch pipe routing based on 3D connection graph and concurrent ant colony optimization algorithm. J Intell Manuf 29:1647–1657. https://doi.org/10.1007/s10845-016-1203-4
54. Liu Q, Wang C (2012) Multi-terminal pipe routing by Steiner minimal tree and particle swarm optimisation. Enterp Inf Syst 6(3):315–327. https://doi.org/10.1080/17517575.2011.594910
55. Liu Q, Wang C (2013) A graph-based pipe routing algorithm in aero-engine rotational space. J Intell Manuf 26. https://doi.org/10.1007/s10845-013-0840-0
56. de Berg M, van Kreveld M, Overmars M et al (2008) Computational geometry: algorithms and applications, 3rd edn. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-77974-2
57. Wormser C (2008) Generalized Voronoi diagrams and applications. PhD thesis, Université Nice Sophia Antipoli. https://hal.science/tel-00410850/
58. Liu Q (2015) A rectilinear pipe routing algorithm: Manhattan visibility graph. Int J Comput Integr Manuf 29:1–10. https://doi.org/10.1080/0951192X.2015.1033019
59. Thantulage G, Kalganova T, Fernando W (2006) A grid-based ant colony algorithm for automatic 3D hose routing. In: 2006 IEEE International Conference on Evolutionary Computation. pp 48–55. https://doi.org/10.1109/CEC.2006.1688289
60. Abd Algfoor Z, Sunar M, Kolivand H (2015) A comprehensive study on pathfinding techniques for robotics and video games. Int J Comput Games Technol 2015:1–11. https://doi.org/10.1155/2015/736138
61. Shin DS, Park BC, Lim CO et al (2020) Pipe routing using reinforcement learning on initial design stage. Journal of the Society of Naval Architects of Korea 57:191–197. https://doi.org/10.3744/SNAK.2020.57.4.191
62. Stanczak M, Pralet C, Vidal V et al (2020) Optimal pipe routing techniques in an obstacle-free 3D space. In: Proceedings of the 9th International Conference on Operations Research and Enterprise Systems - ICORES. pp 69–79. https://doi.org/10.5220/0008923300690079

63. Yin Y, Xu L, Bi Z et al (2013) A novel human-machine collaborative interface for aero-engine pipe routing. IEEE Trans Ind Inf 9:2187–2199. https://doi.org/10.1109/TII.2013.2257805
64. Jiang WY, Lin Y, Chen M et al (2014) An ant colony optimization-genetic algorithm approach for ship pipe route design. Int Shipbuild Prog 61:163–183. https://doi.org/10.3233/ISP-140111
65. Fernando T (2009) Ant colony optimization based simulation of 3D automatic hose/pipe routing. PhD Thesis, Brunel University School of Engineering and Design. http://bura.brunel.ac.uk/handle/2438/4282
66. Liu L, Liu Q (2018) Multi-objective routing of multi-terminal rectilinear pipe in 3D space by MOEA/D and RSMT. In: 2018 3rd International Conference on Advanced Robotics and Mechatronics (ICARM), pp 462–467. https://doi.org/10.1109/ICARM.2018.8610824
67. Ikehira S, Kimura H, Ikezaki E et al (2005) Automatic design for pipe arrangement using multi-objective genetic algorithms. J Jpn Soc Nav Archit Ocean Eng 2:155–160. https://doi.org/10.2534/jjasnaoe.2.155
68. Asmara A, Nienhuis U, Hekkenberg R (2010) Approximate orthogonal simplification of 3D model. In: IEEE Congress on Evolutionary Computation. pp 1–4. https://doi.org/10.1109/CEC.2010.5586162
69. Niu W, Sui H, Niu Y et al (2016) Ship pipe routing design using NSGA-II and coevolutionary algorithm. Math Probl Eng 2016:1–21. https://doi.org/10.1155/2016/7912863
70. Van der Velden C, Bil C, Yu X et al (2007) An intelligent system for automatic layout routing in aerospace design. ISSE 3:117–128. https://doi.org/10.1007/s11334-007-0021-4
71. Martins P, Lobo V (2009) A tool for automatic routing of auxiliary circuits in ships. Paper presented at 11th Seminário Luso-Espanhol de Economia Empresarial 2009. https://cir.nii.ac.jp/crid/1573387450606843904
72. Zhou Q, Lv Y (2020) Research based on Lee algorithm and genetic algorithm of the automatic external pipe routing of the aircraft engine. Int J Mech Eng Appl 8:40. https://doi.org/10.11648/j.ijmea.20200801.16
73. Ajiwaskita F, Gunawan G, Yanuar Y (2020) Pipe-routing optimization using system engineering methodology in ship engine room. p 020022. https://doi.org/10.1063/5.0001001
74. Huibiao L, Peiting S, Yutang Z (2010) Environment modelling for simulation-based pipeline design in engine room. pp 318–320. https://doi.org/10.1109/ICCMS.2010.65
75. Christodoulou S, Ellinas G (2010) Pipe routing through ant colony optimization. J Infrastruct Syst 16:149–159. https://doi.org/10.1061/(ASCE)1076-0342(2010)16:2(149)
76. Rodrigues GPW, Costa LHM, Farias GM et al (2019) A depth-first search algorithm for optimizing the gravity pipe networks layout. Water Resources Management: An International Journal, Published for the European Water Resources Association (EWRA) 33(13):4583–4598. https://doi.org/10.1007/s11269-019-02373-
77. Schmidt-Traub H, Holtkötter T, Lederhose M et al (1999) An approach to plant layout optimization. Chem Eng Technol 22:105–109. https://doi.org/10.1002/(SICI)1521-4125(199902)22:2<105::AID-CEAT105>3.0.CO;2-G
78. Singh J, Cheng JCP (2021) Automating the generation of 3D multiple pipe layout design using BIM and Heuristic search methods. In: Toledo Santos E, Scheer S (eds) Proceedings of the 18th International Conference on Computing in Civil and Building Engineering. Springer International Publishing, pp 54–72. https://doi.org/10.1007/978-3-030-51295-8_6
79. Wu Y, Wang R, Wang Y et al (2018) An area-wide layout design method considering piecewise steam piping and energy loss. Chem Eng Res Des 138:405–417. https://doi.org/10.1016/j.cherd.2018.09.007
80. Sui H, Niu W (2016) Branch-pipe-routing approach for ships using improved genetic algorithm. Front Mech Eng 11:316–323. https://doi.org/10.1007/s11465-016-0384-z
81. Sakti A, Zeidner L, Hadzic T et al (2016) Constraint programming approach for spatial packaging problem. In: Quimper CG (ed) Integration of AI and OR Techniques in Constraint Programming. Springer International Publishing, Cham, pp 319–328. https://doi.org/10.1007/978-3-319-33954-2_23
82. Jiang WY, Lin Y, Chen M et al (2015) A co-evolutionary improved multi-ant colony optimization for ship multiple and branch pipe route design. Ocean Eng 102:63–70. https://doi.org/10.1016/j.oceaneng.2015.04.028
83. Wu BC, Young GS, Schmidt W et al (2009) Applying fuzzy functions and sequential coordination to optimization of machinery arrangement and pipe routing. Nav Eng J 110:43–54. https://doi.org/10.1111/j.1559-3584.1998.tb02964.x
84. Ikehira S, Kimura H (2009) Automatic design algorithm for pipe arrangement considering valve operationality. J Jpn Soc Nav Archit Ocean Eng 9:231–236. https://doi.org/10.2534/jjasnaoe.9.231

85. Bai X, Zhang Y (2012) Engineering rules-based orthogonal and variable-steps pipe routing algorithm for aero-engines. Adv Mater Res 442:104–108. https://doi.org/10.4028/www.scientific.net/AMR.442.104
86. Kang SS, Myung S, Han S (1999) A design expert system for auto-routing of ship pipes. J Ship Prod Des 15. https://doi.org/10.5957/jsp.1999.15.1.1
87. Arora S, Barak B (2009) Computational complexity - a modern approach, 1st edn. Cambridge University Press, Cambridge, United Kingdom. https://doi.org/10.1017/CBO9780511804090
88. Schelbert J (2015) Discrete approaches for optimal routing of high pressure pipes. Doctoral thesis, Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU). https://nbn-resolving.org/urn:nbn:de:bvb:29-opus4-66837
89. Adcock A, Demaine E, Demaine M et al (2014) Zig-Zag Numberlink is NP-Complete. J Inf Process 23. https://doi.org/10.2197/ipsjjip.23.239
90. Lengauer T (1990) Combinatorial algorithms for integrated circuit layout. Applicable Theory in Computer Science. https://doi.org/10.1007/978-3-322-92106-2
91. Lynch J (1975) The equivalence of theorem proving and the interconnection problem. ACM SIGDA Newsletter 5:31–65. https://doi.org/10.1145/1061425.1061430
92. Richards (1984) Complexity of single-layer routing. IEEE Transactions on Computers C-33(3):286–288. https://doi.org/10.1109/TC.1984.1676428
93. Belov G, Cohen L, de la Banda MG et al (2019) Position paper: from multi-agent pathfinding to pipe routing. CoRR abs/1905.08412. https://doi.org/10.48550/arXiv.1905.08412
94. Fortz B, Gouveia L, Joyce-Moniz M (2017) Models for the piecewise linear unsplittable multicommodity flow problems. Eur J Oper Res 261:30–42. https://doi.org/10.1016/j.ejor.2017.01.051
95. Canny J, Reif J (1987) New lower bound techniques for robot motion planning problems. Proc IEEE Conf Foundations Computer Science 49–60. https://doi.org/10.1109/SFCS.1987.42
96. Lozano-Pérez T, Wesley M (1979) An algorithm for planning collision free paths among polyhedral obstacles. Commun ACM 22:560–570. https://doi.org/10.1145/359156.359164
97. Li F, Klette R (2011) Euclidean shortest paths - exact or approximate algorithms. Springer. https://doi.org/10.1007/978-1-4471-2256-2_1
98. Ljubić I (2021) Solving Steiner trees: Recent advances, challenges, and perspectives. Networks 77(2):177–204. https://doi.org/10.1002/net.22005
99. Prömel HJ, Steger A (2002) Geometric steiner problems. Vieweg+Teubner Verlag, Wiesbaden, pp 191–222. https://doi.org/10.1007/978-3-322-80291-0_10
100. Hu Y, Jing T, Hong X et al (2005) An-Oarsman: obstacle-avoiding routing tree construction with good length performance, vol 1. In: Proceedings of the ASP-DAC 2005. Asia and South Pacific Design Automation Conference, 2005. pp 7–12. https://doi.org/10.1109/ASPDAC.2005.1466120
101. Bhagwat A (2015) Obstacle-avoiding rectilinear steiner minimum tree: a survey. International Journal of Innovative Research in Computer and Communication Engineering 3:2810–2816. http://ijircce.com/admin/main/storage/app/pdf/rux2yJw5PkIsV9tab2hrkVJJLqywD1kQoUF6nJmm.pdf
102. Kundu S, Roy S, Mukherjee S (2020) Rectilinear Steiner Tree construction techniques using PB-SAT-based methodology. J Circuits Syst Comput 29(04):2050057. https://doi.org/10.1142/S0218126620500577
103. Kundu S, Roy S, Mukherjee S (2021) An efficient obstacle-avoiding rectilinear Steiner Tree construction method using PB-SAT. IETE J Res 1–11. https://doi.org/10.1080/03772063.2021.1967790
104. Ganley JL, Cohoon JP (1994) Routing a multi-terminal critical net: Steiner tree construction in the presence of obstacles. In: Proceedings of IEEE International Symposium on Circuits and Systems - ISCAS '94, vol 1. pp 113–116. https://doi.org/10.1109/ISCAS.1994.408768
105. Nguyen H, Kim DJ, Gao J (2016) 3D piping route design including branch and elbow using improvements for Dijkstra's Algorithm. Proceedings of the 2016 International Conference on Artificial Intelligence: Technologies and Applications https://doi.org/10.2991/icaita-16.2016.76
106. Bern M (2006) Faster exact algorithms for Steiner trees in planar networks. Networks 20:109–120. https://doi.org/10.1002/net.3230200110
107. Furuholmen M, Glette K, Høvin M et al (2010) Evolutionary approaches to the three-dimensional multi-pipe routing problem: a comparative study using direct encodings. Lect Notes Comput Sci 6022:71–82. In: European Conference on Evolutionary Computation in Combinatorial Optimization (EvoCOP 2010). https://doi.org/10.1007/978-3-642-12139-5_7

108. Jansen R, Vinkesteijn Y, van den Berg D (2020) On the solvability of routing multiple point-to-point paths in Manhattan Meshes. In: Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion. Association for Computing Machinery, New York, NY, USA, GECCO '20, p 1685–1689. https://doi.org/10.1145/3377929.3398098
109. Schmidt-Traub H, Köster M, Holtkötter T et al (1998) Conceptual plant layout. Comput Chem Eng 22:S499–S504. https://doi.org/10.1016/S0098-1354(98)00093-3
110. Kim D, Corne D (1996) Industrial plant pipe-route optimisation with genetic algorithms. Lect Notes Comput Sci 1141:1012–1021. https://doi.org/10.1007/3-540-61723-X_1064
111. Kahng AB, Lienig J, Markov IL et al (2011) VLSI physical design: from graph partitioning to timing closure, 1st edn. Springer Publishing Company, Incorporated. https://doi.org/10.1007/978-90-481-9591-6
112. Dijkstra E (1959) A note on two problems in Connexion with graphs. Numer Math 1:269–271. https://doi.org/10.1007/BF01386390
113. Lee CY (1961) An algorithm for path connections and its applications. IRE Trans Electron Comput 10:346–365. https://doi.org/10.1109/TEC.1961.5219222
114. Ando Y, Kimura H (2011) An automatic piping algorithm including elbows and bends. In: RINA, Royal Institution of Naval Architects - International Conference on Computer Applications in Shipbuilding 2011, Papers. pp 153–158. https://doi.org/10.2534/JJASNAOE.15.219
115. Hart PE, Nilsson NJ, Raphael B (1968) A formal basis for the heuristic determination of minimum cost paths. IEEE Trans Syst Man Cybern 4(2):100–107. https://doi.org/10.1109/TSSC.1968.300136
116. Harabor D, Grastien A (2011) Online graph pruning for pathfinding on grid maps. In: Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence. AAAI Press, p 1114–1119. https://dl.acm.org/doi/10.5555/2900423.2900600
117. Min JG, Ruy WS, Park CS (2020) Faster pipe auto-routing using improved jump point search. International Journal of Naval Architecture and Ocean Engineering 12:596–604. https://doi.org/10.1016/j.ijnaoe.2020.07.004
118. Dorigo M (1992) Optimization, learning and natural algorithms. PhD Thesis, Politecnico di Milano, Italy. https://cir.nii.ac.jp/crid/1573950400977139328
119. Dorigo M, Maniezzo V, Colorni A (1996) Ant system: optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man, and Cybernetics Part B, Cybernetics: a Publication of the IEEE Systems, Man, and Cybernetics Society 26:29–41. https://doi.org/10.1109/3477.484436
120. Fan X, Lin Y, Ji Z (2006) The ant colony optimization for ship pipe route design in 3D space. In: 2006 6th World Congress on Intelligent Control and Automation. pp 3103–3108. https://doi.org/10.1109/WCICA.2006.1712938
121. Stützle T, Hoos HH (2000) Max-min ant system. Futur Gener Comput Syst 16(8):889–914. https://doi.org/10.1016/S0167-739X(00)00043-1
122. Kennedy J, Eberhart R (1995) Particle swarm optimization, vol 4. In: Proceedings of ICNN'95 - International Conference on Neural Networks. pp 1942–1948. https://doi.org/10.1109/ICNN.1995.488968
123. Shi Y, Eberhart RC (1998) Parameter selection in particle swarm optimization. In: Porto VW, Saravanan N, Waagen D et al (eds) Evolutionary Programming VII. Springer Berlin Heidelberg, Berlin, Heidelberg, pp 591–600. https://doi.org/10.1007/BFb0040810
124. Liu Q, Wang C (2010) Pipe-assembly approach for aero-engines by modified particle swarm optimization. Assem Autom 30:365–377. https://doi.org/10.1108/01445151011075825
125. Holland J (1975) Adaptation in natural and artificial systems: an introductory analysis with applications to biology. Control, and Artificial Intelligence. https://doi.org/10.7551/mitpress/1090.001.0001
126. Ito T (2002) Route planning wizard: basic concept and its implementation. Paper presented at the International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, IEA/AIE 2002. https://doi.org/10.1007/3-540-48035-8_53
127. Li H, Zhang Q (2006) A multiobjective differential evolution based on decomposition for multiobjective optimization with variable linkages. In: Parallel Problem Solving from Nature - PPSN IX. Springer Berlin Heidelberg, Berlin, Heidelberg. pp 583–592. https://doi.org/10.1007/11844297_59
128. Huibiao L, Zhefu Y, Peiting S (2009) Hanging bridge algorithm for pipe-routing design in ship engine room. Proceedings - International Conference on Computer Science and Software Engineering, CSSE 1:153–155. https://doi.org/10.1109/CSSE.2008.1216

129. Deliang L, Huibiao L (2009) Interfere-check applying to 3D automatic pipe route arrangement. Proceedings of International Conference on Computational Intelligence and Software Engineering, Wuhan. https://doi.org/10.1109/CISE.2009.5365920
130. Gunawan MDMalik, Asmara LK (2020) Module development of piping system design with rule based algorithms. In: American Institute of Physics Conference Series. p 020019. https://doi.org/10.1063/5.0001000
131. Bai X, Zhang Y (2012) Research on information extraction in pipe-routing layout space for complex products. Appl Mech Mater 155–156:250–254. https://doi.org/10.4028/www.scientific.net/AMM.155-156.250
132. Dong Z, Bian X, Yang S (2020) Space modeling method for ship pipe route design. In: Qin P, Wang H, Sun G et al (eds) Data Science. Springer Singapore, Singapore, pp 384–392. https://doi.org/10.1007/978-981-15-7984-4_28
133. Fan X, Lin Y, Ji Z (2010) The ES model for ship pipes routing design. In: 2010 8th World Congress on Intelligent Control and Automation. pp 2787–2792. https://doi.org/10.1109/WCICA.2010.5554929
134. Liu Q, Xu R, Liu X (2012) A CAD system development for pipe and cable planning. In: 2012 3rd International Conference on System Science, Engineering Design and Manufacturing Informatization. pp 131–133. https://doi.org/10.1109/ICSSEM.2012.6340826
135. Deb K (2001) Multiobjective optimization using evolutionary algorithms. John Wiley & Sons, Ltd, Baffins Lane, Chichester, United Kingdom