To cite this article：LI T L, WANG W S, LIU H Y, et al. Analysis of ship pipeline routing optimization algorithm based on improved artificial bee colony algorithm [J/OL]. Chinese Journal of Ship Research, 2024, 19(2). http://www.ship-research.com/en/article/doi/10.19693/j.issn.1673-3185.03222 (in both Chinese and English).

DOI：10.19693/j.issn.1673-3185. 03222

Analysis of ship pipeline routing optimization algorithm based on improved artificial bee colony algorithm

LI Tieli[1], *WANG Wenshuang*[1], *LIU Haiyang*[2,3], *YANG Yuansong*[2,3], *LIN Yan*[1]

1 School of Naval Architecture Engineering, Dalian University of Technology, Dalian 116024, China

2 CNNC Key Laboratory of Green Construction Technology and Equipment, Beijing 101300, China

3 China Nuclear Industry 23 Construction Co., Ltd, Beijing 101300, China

Abstract: [**Objective**] The artificial bee colony (ABC) algorithm has such characteristics as few control parameters, strong local optimization ability, and fast convergence speed. However, when solving routing optimization problems, it can easily fall into local optimal solutions. In order to solve the problem of pipeline routing in a ship pipeline system, an improved artificial bee colony (IABC) algorithm is proposed. [**Method**] Based on the traditional artificial bee colony algorithm, the crossover operation of genetic operators is introduced into the update mechanism of following bees, and an adaptive strategy is adopted for the crossover probability of the crossover operator. The crossover operation on the population is used to find new solutions in the global range. The way scout bees search for new paths is improved from updating the points that the path passes to updating the "road sections" in the path. This paper proposes an artificial bee colony coevolutionary algorithm for solving the optimization of branch pipeline paths. [**Results**] Compared with the standard artificial bee colony algorithm, the improved algorithm can improve the path layout effect by 32.3%−37.4% and the convergence speed by 17.7%−29.9%. [**Conclusion**] The improved artificial bee colony algorithm proposed herein has higher solution quality, faster convergence speed, and better stability than the traditional artificial bee colony algorithm for a single pipe or branch pipe.

Keywords: ship pipeline; artificial bee colony (ABC) algorithm; path planning; co-evolutionary algorithm

CLC number：U664.84

0 Introduction

The layout design of the ship's engine room is an important part of the overall layout design. Similar to the overall design of the ship, engine room design is also an iterative, spiral process with continuous updates and refinements. The engine room contains numerous equipment and complex structures, paired with various types of pipelines, including both single and branch pipelines. In general, the structure, the equipment, and the piping system are interdependent and influence each other. In conventional design processes, designers must repeatedly modify and adjust their work to achieve the final solution, which consumes considerable time and labor. Swarm intelligence optimization algorithms, as one of the most common modern optimization methods, are independent of the analytical nature of the problem and possess parallelism characteristics. These algorithms can significantly facilitate pipeline routing design, serving as a supplement to conventional design approaches.

The artificial bee colony (ABC) algorithm is a heuristic algorithm that simulates the foraging behavior of natural bee populations. It exemplifies swarm intelligence and is distinguished by having few control parameters, independence from specific problem details, robust local optimization capabilities, and rapid convergence speed. It is an effective method for solving multi-objective optimization problems and optimizing nonlinear continuous functions.

Researchers worldwide have conducted extensive research on the ABC algorithm and pipeline

Received：2022－12－15 **Accepted**：2023－03－28

***Corresponding author**：LIN Yan

optimization design. The ABC algorithm was first proposed by Karaboga[1] in 2005 to optimize algebraic problems. Wang et al. [2] proposed a path planning method based on an improved bee colony algorithm using cubic Bezier curve optimization, effectively converting the path planning problem into an optimization problem for generating Bezier curve control points, thereby facilitating effective collision-free path planning for smooth paths. Zhang et al. [3] introduced the ABC algorithm to address path planning problems for aircraft engine pipelines, developing an improved multi-objective ABC algorithm for intelligent pipeline layout in aircraft engines, thereby achieving diversity and intelligence in the layout. He et al. [4] used the maximum and minimum distance product to enhance the ABC algorithm, addressing issues of slow convergence and prematurity in the algorithm's later phases, and implemented it in path planning of unmanned aerial vehicle (UAV). Li et al. [5] introduced adaptive selection, crossover, and mutation operations in genetic operators into ABC algorithm, improving convergence speed and the solution quality when solving the warehouse robot path planning. Luo et al. [6] proposed a midship section structure optimization method based on parametric geometric modeling analysis and ABC algorithm to solve challenges in considering the profile quantity changes during section structure optimization. This approach was validated by optimizing the section structure of an actual ship subjected to the total longitudinal bending moment. In terms of ship pipeline routing optimization, Lin et al. [7-10] designed an improved particle swarm algorithm based on nonlinear adaptive inertia weight for optimizing the pipeline layout of the nuclear power primary loop system. Compared with the standard particle swarm algorithm, this improved algorithm enhances convergence speed and mitigates the issue of local optima. By combining it with a coevolutionary algorithm, it can realize the collaborative layout optimization of branch pipelines. Bian et al. [11-12] proposed an improved A* algorithm and genetic algorithm for path planning of ship pipelines. They designed several novel A*-GA genetic operators and established an algorithm framework for calculating single, branch, and multiple pipelines. The feasibility and effectiveness of the algorithms were verified through experimental validation.

In routing optimization calculation for branch pipelines, researchers tend to employ the concept of coevolutionary algorithm. Polat et al. [13] introduced coevolutionary algorithms to address vehicle routing problems, which markedly improved the solution quality compared with the genetic algorithm. In addressing the optimization problem of ship branch pipeline layout, Wu [14] introduced a novel approach based on coevolutionary algorithm. This method involves breaking down the branch pipeline system into several individual pipelines, each represented by a population. These populations influence each other to achieve the optimal layout of the branch pipeline system. Dong et al. [15] tackled the coordinated layout problem of ship pipelines by proposing an algorithm framework for multiple or branch pipeline layouts based on an improved particle swarm algorithm. Jiang et al. [16] introduced a coevolutionary improved multi-ant colony optimization (CIMACO) algorithm for multiple pipeline optimization design in ships. This method demonstrated enhanced performance in circumventing local optima and accelerating convergence speed.

Classic optimization algorithms, such as the genetic and particle swarm algorithms, are extensively applied in ship and aircraft engine pipeline layouts, as mentioned in the references above. In recent years, the ABC algorithm has emerged to address non-deterministic and multi-objective challenges, garnering significant interest. Its application predominantly involves 2D robot path planning and optimizing pipeline layouts using simplistic 3D models. However, the ABC algorithm is unsuitable for optimizing the routing of branch pipelines. Moreover, the conventional ABC algorithms are prone to local optima, and their optimization efficacy tends to diminish in the latter phases of the process. There is considerable potential for improvement, particularly in pipeline layout optimization. Pathfinding and algorithmic logic improvements could substantially elevate the solution quality and accelerate convergence speed.

This paper addresses the challenge of pipeline path planning within ship pipeline systems, focusing primarily on refining the computational process to achieve the optimal pipeline layout. ABC algorithm is utilized to tackle the optimization issue, and improvements are made to address the shortcomings of the basic ABC algorithm. Concurrently, by integrating the concept of coevolutionary algorithm, an optimization algorithm tailored for branch pipeline routing optimization is introduced. Ultimately, the feasibility and efficiency of the algorithm are verified through actual engineering cases.

1 Description of pipeline path planning problems

The pipeline routing optimization problem, which involves identifying the optimal pipeline layout within a specified layout space under certain conditions, constitutes a complex, constrained multi-objective optimization problem, as expressed in Eq. (1):

$$\begin{cases} \min F(x) = (f_1(x), f_2(x), \ldots, f_m(x)) \\ \text{s.t} \begin{cases} g_i(x) \leqslant 0, & i = 1,2,\ldots,k \\ h_j(x) = 0, & j = 1,2,\ldots,q \\ L_\mathrm{b} \leqslant x \leqslant U_\mathrm{b} \end{cases} \end{cases} \tag{1}$$

where x represents the design variable; $f(x)$, $g_i(x)$, and $h_j(x)$ represent the objective function, inequality constraint, and equality constraint of the pipeline path optimization problem, respectively; m, k, and q represent the number of the objective function, inequality constraint, and equality constraint, respectively; L_b and U_b are the lower and upper bounds of the design variable values.

1.1 Coding method

For the pipeline path planning problem, each solution is represented by the expression of a single path. This paper utilizes a fixed-length coding approach, where the basic unit comprises the 3D coordinates of the nodes along the path. Consequently, the encoding method for a path is expressed as Eq. (2):

$$path_i = [(x_\mathrm{s}, y_\mathrm{s}, z_\mathrm{s}), (x_1, y_1, z_1), (x_2, y_2, z_2), \ldots, (x_\mathrm{e}, y_\mathrm{e}, z_\mathrm{e})] \tag{2}$$

where $(x_\mathrm{s}, y_\mathrm{s}, z_\mathrm{s})$ and $(x_\mathrm{e}, y_\mathrm{e}, z_\mathrm{e})$ represent the 3D coordinates of the start and end points of the pipeline, respectively, with the remaining points denoting the nodes through which the pipeline path traverses.

1.2 Objective function

The objective function for the routing optimization problem studied in this paper primarily considers the following five aspects.

1) Path length function.

$$L(x) = \sum_{i=1}^{n} l_i \tag{3}$$

where $L(x)$ represents the total length of the pipeline path; l_i represents the length of the i-th road section within the pipeline; n represents the number of road sections in the pipeline.

2) Elbow number function.

The elbow number function, denoted by $B(x)$, represents the number of elbows in a path. When determining whether a specific intermediate node in the path forms an elbow, the positions of the two adjacent nodes are considered. If the three nodes are collinear, the intermediate node is not classified as an elbow; otherwise, it is. Assuming that p is the node to be evaluated, with its preceding and succeeding nodes defined as, p_1 and p_2 respectively. This paper employs Eqs. (4) and (5) to calculate the unit vector $\boldsymbol{e}_1$ between p_1 and p_2 and the unit vector $\boldsymbol{e}_2$ between p and p_2, respectively.

$$\boldsymbol{e}_1 = \frac{1}{\sqrt{(x-x_1)^2 + (y-y_1)^2 + (z-z_1)^2}} (x-x_1, y-y_1, z-z_1) \tag{4}$$

$$\boldsymbol{e}_2 = \frac{1}{\sqrt{(x_2-x)^2 + (y_2-y)^2 + (z_2-z)^2}} (x_2-x, y_2-y, z_2-z) \tag{5}$$

where (x, y, z) are the 3D coordinates of point p; (x_1, y_1, z_1) and (x_2, y_2, z_2) are the 3D coordinates of point p_1 and p_2, respectively.

Elbow number function is calculated by Eq. (6).

$$\begin{cases} B(x) = \sum_{i=1}^{n_\mathrm{p}-2} p_i \\ p_i = \begin{cases} 1, & \boldsymbol{e}_1 \neq \boldsymbol{e}_2 \\ 0, & \boldsymbol{e}_1 = \boldsymbol{e}_2 \end{cases} \end{cases} \tag{6}$$

where n_p is the number of nodes in the path.

3) Orthogonal function.

$$O(x) = \sum_{i=1}^{n-1} \sum \min \text{ two of } \{d_x, d_y, d_z\} \tag{7}$$

where $O(x)$ represents the path orthogonality value, if $O(x) = 0$, the paths are entirely orthogonal; the variables d_x, d_y, and d_z are the absolute differences in the x, y, and z coordinates between the i-th node and the $(i+1)$-th node, respectively; $\sum\min$ to of $\{d_x, d_y, d_z\}$ is the minimum sum of the two minimum values among d_x, d_y, and d_z .

4) Energy area function.

The energy area function is represented by $E(x)$. Within the 3D space, pipeline layout is encouraged to be placed around obstacles and in the boundary regions of the map. These regions are set as energy areas. The energy area around obstacles is set as shown in Fig. 1. The energy area function denotes the length of the pipeline routing traversing the energy area, as depicted in Eq. (8). A larger calculated result indicates that the pipeline traverses more energy areas, signifying a better layout effect.

$$E(x) = \sum_{i=1}^{n} l_e \tag{8}$$

where l_e is the path length through the energy area in the *i*-th road section.

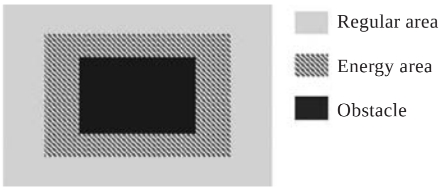


Fig. 1 Schematic diagram of energy area

5) Penalty function.

The penalty function is represented by $P(x)$, which penalizes unreasonable situations in the path, such as interference with obstacles or path retracing.

By integrating the objective functions expressed in Eqs. (1) − (5), we derive the overall objective function for the pipeline optimization problem, as shown in Eq. (9). This is also called fitness function.

$$F(x) = a \cdot L(x) + b \cdot B(x) + c \cdot O(x) - d \cdot E(x) + e \cdot P(x) \tag{9}$$

where *a*, *b*, *c*, *d*, and *e* are weight coefficients used to control the weight proportions of the sub-objective functions $L(x)$, $B(x)$, $O(x)$, $E(x)$, and $P(x)$ in the algorithm.

From the optimization problem expression Eq. (1) and the overall objective function Eq. (9), it is clear that this study transforms the multi-objective optimization into the single-objective optimization of pipeline routing. This means it is converted into a minimization problem, where a smaller $F(x)$ value indicates a better pipeline layout. Due to the different units and significant numerical discrepancies in the calculation results of each sub-objective function, to more reasonably represent each sub-objective function in the optimization process, some sub-objective functions are non-dimensionalized using Eq. (10), making their values fall within the range of [0, 1].

$$x^* = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \tag{10}$$

where $x_{\min}$ and $x_{\max}$ are the minimum and maximum values of the objective function results.

A summary of the dimensionalization methods for the sub-objective functions is provided in Table 1.

1.3 Constraints

Pipeline layout problems are similar to path planning problems but have distinct differences. The ship pipeline layout needs to consider the influence of multiple constraints, including physical, economic, production, and safety constraints, making it a complex pipeline path planning problem with numerous constraints. The specific constraints include the followings:

1) Connectivity between pipeline start and end points. Connectivity is a physical constraint of pipeline layout, and the path of the pipeline should ensure continuity between the start and end points without any interruptions.

2) Non-interference between the pipeline and obstacles. Assuming the pipeline passes through *n* path nodes and there are *m* obstacles in the layout space, to meet the physical constraint of avoiding interference between the pipeline and obstacles, the situation described in Eq. (11) must be prevented.

$$\begin{cases} X_{j\min} \leqslant x_i \leqslant X_{j\max}, \ i = 1, 2, \ldots, n, \ j = 1, 2, \ldots, m \\ Y_{j\min} \leqslant y_i \leqslant Y_{j\max}, \ i = 1, 2, \ldots, n, \ j = 1, 2, \ldots, m \\ Z_{j\min} \leqslant z_i \leqslant Z_{j\max}, \ i = 1, 2, \ldots, n, \ j = 1, 2, \ldots, m \end{cases} \tag{11}$$

where x_i, y_i, and z_i are the 3D coordinates of the *i*-th path node; $X_{j\min}$ and $X_{j\max}$, $Y_{j\min}$ and $Y_{j\max}$, $Z_{j\min}$ and $Z_{j\max}$ represent the minimum and maximum values of the *j*-th obstacle's coordinate range in the *x*, *y*, and *z* directions, respectively, within the layout space.

Table 1 Dimensionalization method and description of each sub-objective function

Sub-objective function	Value description	
Path length function $L(x)$	$x^* = \frac{x - x_{\min}}{x_{\max} - x_{\min}}$	$x_{\min}$ is represented by the straight-line distance between the starting points of the pipeline. $x_{\max}$ is represented by the maximum value of all path lengths in the initial population.
Elbow number function $B(x)$	$x^* = \frac{x - x_{\min}}{x_{\max} - x_{\min}}$	$x_{\min}$ is set to 0, indicating that the path has no elbows. $x_{\max}$ is represented by the maximum value of the elbow number inm in all paths in the initial population.
Orthogonal function $O(x)$	$x^* = \frac{x - x_{\min}}{x_{\max} - x_{\min}}$	$x_{\min}$ is set to 0, indicating that the path is entirely orthogonal. $x_{\max}$ is represented by the maximum orthogonality value of all paths in the initial population.
Energy area function $E(x)$	$x^* = \frac{E(x)}{L}$	This equation represents the ratio of the path length that passes through the energy area to the total path length, where L is the total path length.
Penalty function $P(x)$	$x^* = \begin{cases} 0, & P(x) = 0 \\ 1, & P(x) \neq 0 \end{cases}$	If there are any unreasonable conditions in the path, the penalty function after dimensionalization is set to 1; otherwise, it is set to 0.

3) Minimize path length. The goal is to satisfy economic constraints and reduce the cost of pipeline layout.

4) Minimize the elbow number. The goal is also to satisfy economic constraints and reduce the cost of pipeline layout.

5) Ensure orthogonality of the pipeline layout. The pipelines should be positioned orthogonally within the layout space to meet production constraints, avoiding oblique lines.

6) Maximize pipeline placement in energy areas. In practice, most pipelines are preferred to be placed close to equipment or bulkheads to meet production and safety constraints. However, certain pipelines, such as those for fuel oil transport, should avoid being routed above boilers. These constraints are managed by setting energy areas.

2 ABC algorithm improvements and its application in the concept of coevolution

2.1 Basic ABC algorithm

ABC algorithm primarily consists of four basic elements: food sources, leading bees, following bees, and scout bees. These three types of bees collaborate and share information about the food sources during the foraging process (searching process), thereby finding the optimal solution to the problem.

1) Food sources correspond to the solutions of the problem, with the quality of the food source representing the quality of the solution.

2) Leading or employed bees are tasked with locating food sources. Consequently, their numbers are proportional to the food sources. In each iteration, a leading bee searches within its neighborhood. If a new solution is better, it will replace the old one; otherwise, the old one is retained. Furthermore, leading bees share information about the food sources for other bees to select and evaluate.

3) Following bees select food sources based on the information provided by leading bees, generally using a roulette wheel selection method. They are also responsible for sharing information about food sources, allowing other bees to make informed choices and judgments.

4) Scout bees help avoid local optima. If a leading bee fails to find a better solution after several iterations in the iterative process, it will become a scout bee and search for a new food source globally.

The calculation process of the ABC algorithm is illustrated in Fig. 2.

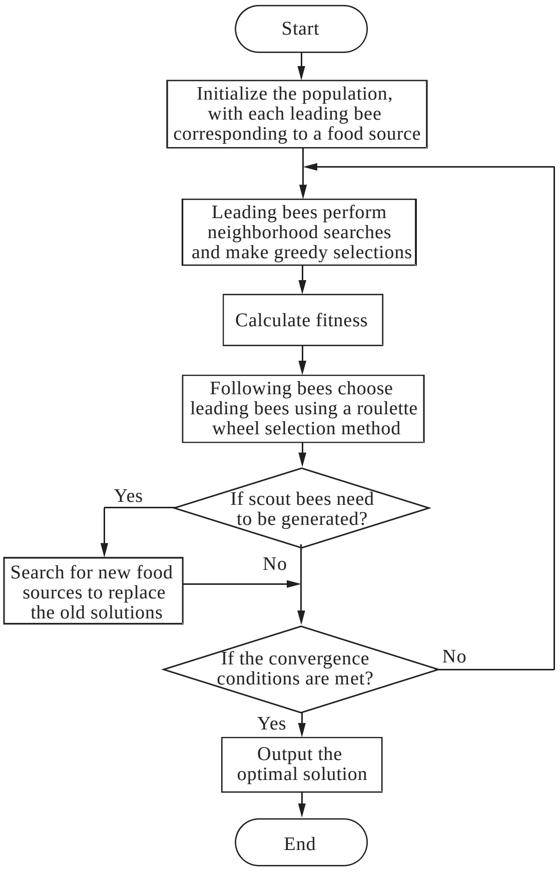

Start
Initialize the population, with each leading bee corresponding to a food source
Leading bees perform neighborhood searches and make greedy selections
Calculate fitness
Following bees choose leading bees using a roulette wheel selection method
If scout bees need to be generated?
Yes
No
Search for new food sources to replace the old solutions
If the convergence conditions are met?
No
Yes
Output the optimal solution
End

Fig. 2 Flow chart of ABC algorithm

The main calculation steps of ABC algorithm are as follows:

1) In the population initialization phase, first set the number of food sources to N, which is also the number of leading bees N. The dimension of the solution to the problem is D. The initial population of leading bees is generated according to Eq. (12).

$$x_{ij} = x_{j\min} + \delta \cdot (x_{j\max} - x_{j\min}) \tag{12}$$

where $i \in [1, N]$; $j \in [1, D]$; $x_{j\max}$ and $x_{j\min}$ represent the respective upper and lower bounds of the j-th dimension of the solution; δ denotes a random number within the range [0, 1].

2) In the leading bee phase, during each iteration, each leading bee conducts a neighborhood search using Eq. (13) to obtain a new candidate solution z_{ij}. The fitness of both the old solution x_{ij} and the new solution z_{ij} is then calculated and evaluated, with the better solution selected for the next iteration.

$$z_{ij} = x_{ij} + \phi_{ij}(x_{ij} - x_{dj}) \tag{13}$$

where $i \in [1, N]$, $d \in [1, N]$, and $d \neq i$, $j \in [1, D]$; ϕ_{ij}

is a random number in the range [−1, 1].

3) In the following bees phase, after the leading bees finish searching and updating, they will share the solution information with the following bees. The following bees select the leading bees based on the quality of the solution, typically using the roulette wheel selection method. The probability P_i of selecting a particular leading bee is calculated using Eq. (14). Meanwhile, after selecting a leading bee to follow, the following bees will also perform a neighborhood search.

$$P_i = \frac{fit_i}{\sum_{k=1}^{N} fit_k} \tag{14}$$

where fit_i is the fitness value of the *i*-th solution.

4) In the scout bee phase, if a food source x_i has not been updated for a given number of iterations (*limit*), it is deemed that the solution is trapped in a local optimal solution, and the food source will be abandoned. The leading bees will then become scout bees, generating a new solution to replace the old one, according to Eq. (12).

2.2 Improvements to the ABC algorithm

2.2.1 Including mutation operation in genetic operators

In ABC algorithm, the role of following bees is to selectively follow the path information shared by leading bees, typically using a roulette wheel selection method. After selection, they perform a neighborhood search to find potentially better solutions, which promotes convergence and enhances diversity within the neighborhood. But this neighborhood search is similar to the update method of leading bees, which serves as a supplementary role and lacks the function of generating new solutions globally. In contrast, the crossover method in genetic operators is primarily used to obtain new solutions globally. Introducing the crossover strategy into the update method of following bees can not only ensure that leading bees continue to update better solutions within the neighborhood, but also allow the entire population to supplement new solutions globally.

Therefore, based on the conventional ABC algorithm, the optimization strategy of following bees can be modified as follows: first, randomly select two paths shared by leading bees and use a single-point crossing method to determine the position of the crossing point in the paths. Then, exchange the path information after the crossing point to obtain two new paths, as shown in Fig. 3. Finally, calculate the fitness of the two new paths, compare them with the corresponding following bees from the previous generation, and perform a greedy selection.

Parental generation 1=[[62, 50, 21], [62, 50, 21], [20, $\underline{32}$, 56], [9, 17, 63], [9, 14, 75]]
Parental generation 2=[[62, 50, 21], [62, 34, 21], [19, $\underline{24}$, 33], [30, 49, 75], [9,14, 75]]

Δ
Crossing point

Filial generation 1=[[62, 50, 21], [62, 50, 21], [20, 32, 33], [30, 49, 75], [9, 14, 75]]
Filial generation 2=[[62, 50, 21], [62, 34, 21], [19, 24, 56], [9, 17, 63], [9,14, 75]]

Fig. 3 Schematic diagram of single point crossing

2.2.2 Implement adaptive strategies for cross-over operators

In the basic genetic algorithm, the crossing probability P_c remains constant. A higher crossing probability P_c can ensure more high-quality solutions in the early phases of population evolution. However, in the later phase of population evolution, a higher crossing probability P_c can destroy excellent individuals. Therefore, this paper proposes a crossing probability P_c that can change according to the fitness differences between population groups. The calculation method for adaptive crossing probability P_c is shown in Eq. (15):

$$P_\mathrm{c} = \begin{cases} P'_\mathrm{c} + (P_\mathrm{cmax} - P'_c) \times \cos\left(\dfrac{f_\mathrm{a} - f'}{f_\mathrm{a} - f_\mathrm{min}} \times \dfrac{\pi}{2}\right), & f' < f_\mathrm{a} \\ P'_\mathrm{c} - (P'_\mathrm{c} - P_\mathrm{cmin}) \times \sin\left(\dfrac{f' - f_\mathrm{a}}{f_\mathrm{max} - f_\mathrm{a}} \times \dfrac{\pi}{2}\right), & f' \geqslant f_\mathrm{a} \end{cases} \tag{15}$$

where P'_c is the crossing probability parameter; P_cmax and P_cmin are the upper and lower bounds of crossing probability; f_max and f_min are the maximum and minimum fitness values of the current population; f_a is the average fitness value of the current population; f' is the larger fitness value among the two crossover individuals.

The function of Eq. (15) is to adaptively modify the crossing probability based on the characteristics of population fitness. After introducing the crossing operator into the basic ABC algorithm, a larger value of the crossing operator can improve the ability of the algorithm to search for new solutions globally in the early phases of the algorithm. However, as the algorithm progresses, the fitness of each individual in the population tends to stabilize. Through Eq. (15), the crossing probability can be reduced to prevent the destruction of the better individuals in the population.

2.2.3 Enhancing the strategy of scout bees searching for new paths

For the strategy of finding new paths after the scout bees are activated, this paper proposes improvements to ABC algorithm from two aspects. Firstly, the scout bee is generated when leading bees have not found a better solution after a certain amount of iterations (*limit*). At this point, the solution found by the leading bee might be a global optimal solution or trapped in a local optimal solution. If the solution is a global optimal solution, generating the scout bee will destroy this optimal solution. Therefore, an elite retention strategy can be introduced, where the fitness of all individuals in the population is assessed before generating the scout bees. If the best individual in the population is found, it is retained for the next generation; otherwise, a new solution continues to be sought. Additionally, for the specific path optimization problem, the solution can be represented by multiple continuous segments from the starting point to the endpoint of the path. In addition to being represented by "elbow points", it can also be represented by "road sections". Leading bees iteratively optimize the coordinates of the "elbow points" to find the optimal path. In the later phase of the algorithm, there may be a situation as shown in Fig. 4. When the path is trapped in a local optimal solution, the global optimal solution can only be obtained by updating P_2 to the position of P_{best} while keeping other points unchanged. At this point, it becomes difficult to avoid local optimal solution or discover a better path through neighborhood search by leading bees or re-search by scout bees. This difficulty mainly arises because the solution's dimensionality is overly complex when optimizing by updating points, and the update direction is excessively random. Therefore, this paper proposes a new update strategy for scout bees, shifting the focus of the path update to "road sections". Since the start point of the path is fixed, in a path with 5 nodes, only 2 "road sections" can be iteratively updated. The update direction of the "road section" is limited to the direction perpendicular to the section, significantly increasing the probability of avoiding local optima.

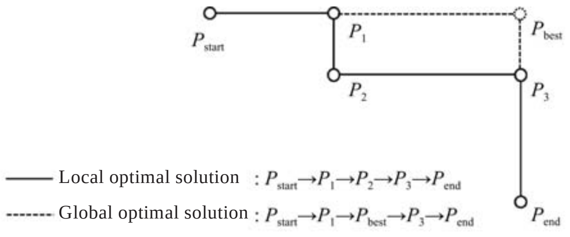

P_{start}
P_1
P_{best}
P_2
P_3
P_{end}
Local optimal solution : $P_{\text{start}}\rightarrow P_1\rightarrow P_2\rightarrow P_3\rightarrow P_{\text{end}}$
Global optimal solution : $P_{\text{start}}\rightarrow P_1\rightarrow P_{\text{best}}\rightarrow P_3\rightarrow P_{\text{end}}$

Fig. 4 Local optimal solution and global optimal solution

After improving the algorithm, scout bees adopt a new path updating strategy, as shown in Fig. 5. In the optimization process, road section P_1P_2 can be updated first. Since the optimization can only be performed in the direction perpendicular to the road section, points P_1 and P_2 have a greater probability of being updated to the positions of P_{best} and P_3. The road section P_2P_3 can also be updated. Points P_2 and P_3 are more likely to be updated to P_1 and P_{best}. Both update approaches can achieve the global optimal solution. The path continues to be represented by node coordinates. After the final result is obtained, the path is simplified by removing redundant nodes, resulting in the final optimal path:$[(P_{\text{start}_x}, P_{\text{start}_y})]$, $[(P_{\text{best}_x}, P_{\text{best}_y})]$, $[(P_{\text{end}_x}, P_{\text{end}_y})]$.

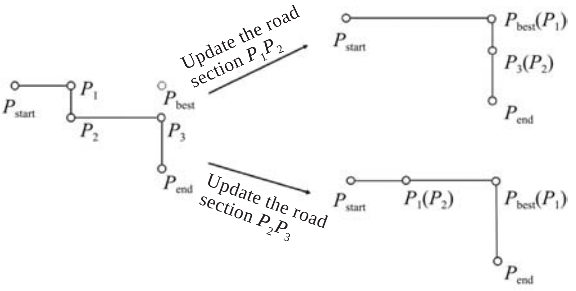


Fig. 5 A strategy for road section optimization

2.3 Optimization design of branch pipeline routing using ABC coevolution algorithm

If the routing optimization of a single pipeline is considered as a single objective, then the branch pipeline optimization can be viewed as a problem of simultaneously optimizing multiple objectives. The concept of coevolution can be described as follows: A system consists of multiple subsystems or can be divided into several subsystems. While independently completing their evolutionary processes, these subsystems influence each other. They are capable of "sharing" information. As each subsystem evolves, it exchanges beneficial information with others, enabling the entire system to develop collaboratively and ultimately find the optimal solution for the whole system.

Based on the concept of coevolution, this paper proposes an optimization strategy adapted to the ABC algorithm, specifically designed to address the optimization challenge of branch pipeline routing. Typically, a network of branch pipelines includes a main pipeline, with other pipelines branching off from it at a junction point, as illustrated in Fig. 6. The pipeline from P_{start} to P_{end1} represents the main pipeline, while the pipeline from P_{start} to P_{end2} is the branch pipeline. Since the starting point of the branch pipeline must be on the main pipeline, the routing optimization problem of branch pipeline can be transformed into two tasks: determining the exact location of the branch point on the main pipeline and finding the optimal path from the branch point to the endpoint of the branch pipeline. In this paper, the location of the branch point along the main pipeline is represented by a decimal r, where $r \in [0, 1]$. When $r = 0$, the branch point is at the start of the main pipeline; when $r =1$, it is at the end of the main pipeline. Additionally, r can represent the ratio of the Manhattan distance from the start of the main pipeline to the branch point, relative to the Manhattan distance between the start and end points of the main pipeline. For instance, in Fig. 6, the branch point location is $r = 0.43$. During the population initialization, the branch point positions are generated randomly. Throughout the iterative optimization process, these positions are updated according to the optimization method used by the leading bees in the ABC algorithm (Eq. (13)).

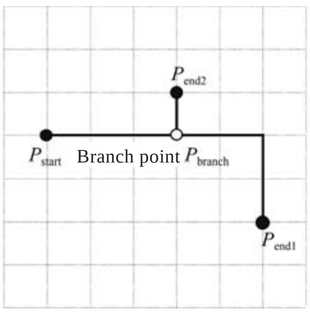


Fig. 6 Example diagram of branch pipeline

Thus, the optimization problem of branch pipeline paths can be divided into three tasks: the single pipeline routing optimization of the main pipeline, the determination of branch point positions for other pipelines, and the single path optimization from the branch point to the endpoint after the branch point positions are determined. The key lies in applying the concept of coevolution to establish connections between these tasks, thereby achieving coevolution.

The basic flowchart for branch pipeline routing optimization based on the ABC algorithm is illustrated in Fig. 7. After the algorithm is initiated, the main pipeline for the branch network is first established based on the start point location of each branch path, and the initial population for the main pipeline is generated. Next, the branch points for other pipelines, which serve as their starting points, are randomly determined on each individual of the main pipeline's population. The initial populations for these branch pipelines are then generated. The iterative process follows, with each iteration including updates to the main pipeline paths, branch point positions, and other pipeline paths. Finally, after the iterations conclude, the optimal paths for all pipelines are output.

3 Simulation example of ship pipeline routing optimization based on improved ABC algorithm

3.1 Experimental verification and simulation example of single pipeline routing optimization

To verify the feasibility and efficiency of the improved artificial bee colony (IABC) algorithm proposed in this paper, it is necessary to compare the experimental results with those from relevant studies on pipeline routing optimization. Therefore, two numerical simulation examples of single pipeline routing from Ref. [17] are selected, and the proposed IABC algorithm is applied to these cases. The results are then compared with those reported in the reference.

First, the computational environment must be specified. All calculations in this section are performed on a computer running the Windows10 operating system, equipped with an AMD Ryzen 54600H processor and 16 GB of memory. The optimization algorithm is written in Python3.7 using the PyCharm 2021 software.

For the single pipeline calculation, the parameters

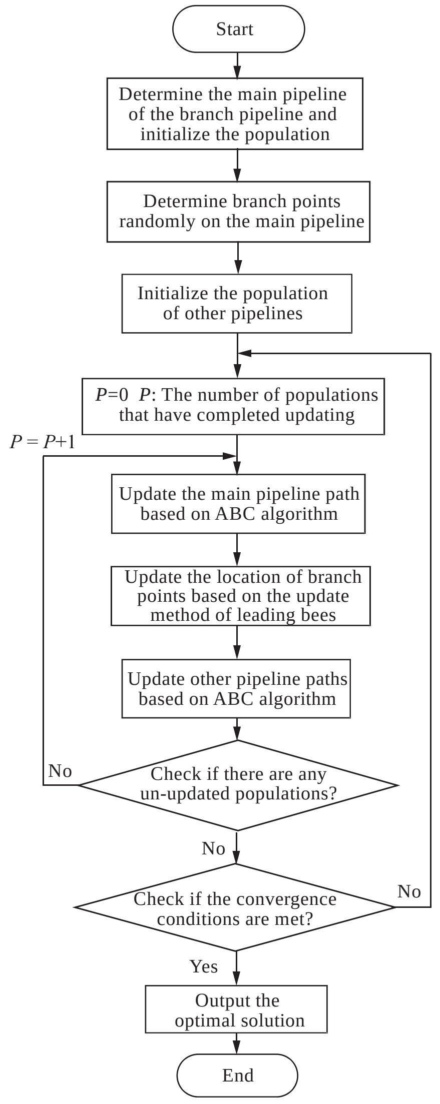
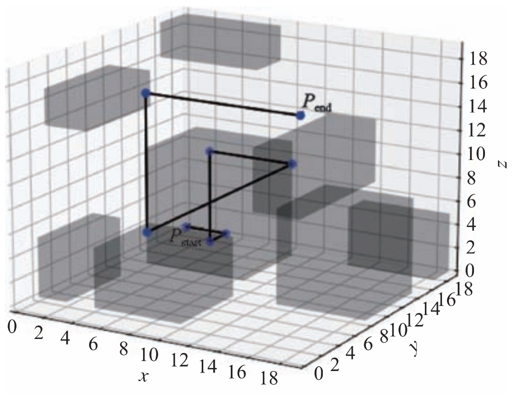

Start
Determine the main pipeline of the branch pipeline and initialize the population
Determine branch points randomly on the main pipeline
Initialize the population of other pipelines
P=0 P: The number of populations that have completed updating
P = P+1
Update the main pipeline path based on ABC algorithm
Update the location of branch points based on the update method of leading bees
Update other pipeline paths based on ABC algorithm
No
Check if there are any un-updated populations?
No
Check if the convergence conditions are met?
No
Yes
Output the optimal solution
End

Fig. 7 Flow chart of branch pipeline routing optimization based on ABC coevolutionary algorithm

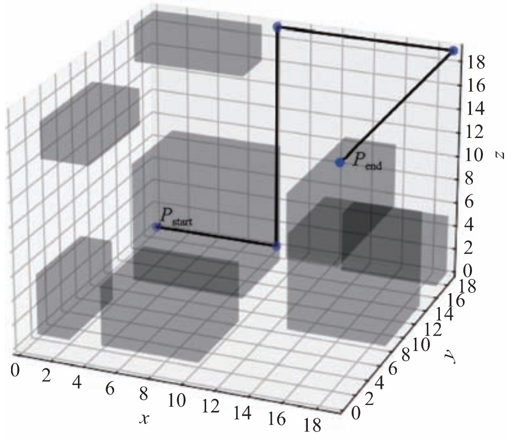
of the IABC algorithm are set as follows: the number of food sources (equivalent to the number of leading bees) N = 30, the number of following bees N_o = 30, the maximum number of iterations Max_G = 200, the maximum cycle number of a food source Max_$Limit$ = 15, and crossing probability parameter P_c' = 0.7, with the upper and lower bounds set to P_{cmax} = 0.9 and P_{cmin} = 0.5 respectively. If the weights of each sub-objective function are all set to 1, there is a tendency for the path to follow an oblique line, as the path should be shorter with fewer elbows. Therefore, when setting the weights of each sub-objective function, it is necessary to decrease the weights for path length and elbow count while increasing the orthogonality weight. The weight coefficients are set to a = 0.8, b = 0.8, c = 2, d = 1, e = 1.

Using the IABC algorithm introduced in this paper, computations were conducted on models 2 and 6 from Ref. [17]. The results obtained from these examples are compared with those in the reference, as shown in Figs. 8 and 9, respectively. A comparison of average convergence iterations, path lengths, and other relevant data is summarized in Table 2. The results indicate that the IABC algorithm achieves satisfactory path optimization and demonstrates fast convergence for single pipeline routing.

Furthermore, to verify that the IABC algorithm can efficiently solve the actual layout problem of single pipelines on ships, this paper uses the fuel oil

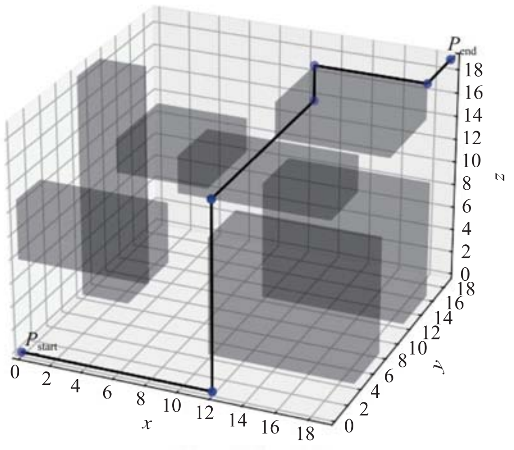


(a) Calculation results of algorithm in Ref. [17]

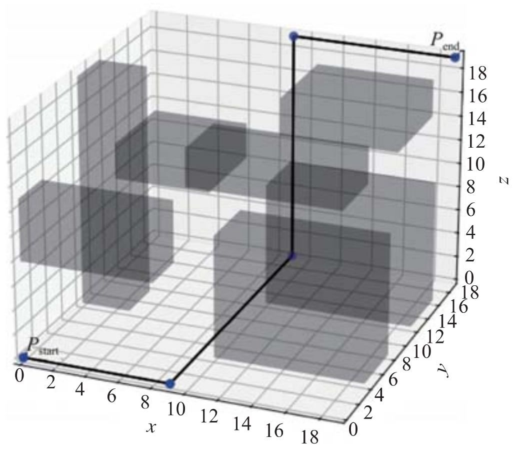


(b) Calculation results of improved ABC algorithm

Fig. 8 Calculation results of single pipeline of model 2 in Ref. [17]



(a) Calculation results of algorithm in Ref. [17]



(b) Calculation results of improved IABC algorithm

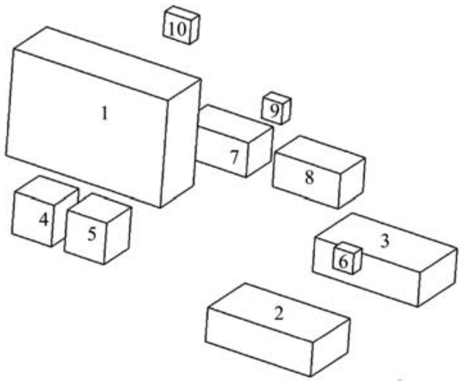
Fig. 9 Calculation results of single pipeline of model 6 in Ref. [17]

piping system in the ship's engine room as the research object and conducts a routing optimization calculation for the single pipelines involved. The required equipment for the fuel oil piping system is simplified according to their shapes, and the equipment is arranged in the corresponding positions within the 3D layout space of the engine room, which measures 15 000 mm in length, 12 000 mm in width, and 8 000 mm in height. The simplified layout space of the ship's fuel oil piping system is shown in Fig. 10, and the positions of various equipment in the space can be expressed by the diagonal coordinates of each equipment, as shown in Table 3.

Table 2 Data comparison of single pipeline results of two models calculated by IABC algorithm and the algorithm in Ref. [17]

Parameter	Model 2			Model 6		
	Algorithm in Ref.[17]	IABC algorithm	Reduction rate/%	Algorithm in Ref.[17]	IABC algorithm	Reduction rate/%
Average convergence generations	41.2	26.8	35.0	90.8	28.7	68.4
Average convergence time/s	6.4	4.1	35.9	11.8	4.3	63.4
Path length/mm	5 700	5 700	0	5 700	5 700	0
Number of elbows	6	3	50	5	3	40

The pipeline extending from the oil purifier supply pump to oil purifier 1 was selected for the single pipeline simulation experiment. The coordinates for the start and end points of this pipeline are (11 800, 6 000, 2 000) and (4 800, 2 700, 2 800), respectively. Routing optimization was performed using the ABC and IABC algorithms, with 20 iterations each. As shown in Table 4, the results indicate that both algorithms achieved satisfactory path layout results, and the optimal layout solutions obtained by both algorithms were identical. The optimal pipeline path is: [(11 800, 6 000, 2 000), (11 800, 2 700, 2 000), (4 800, 2 700, 2 000), (4 800, 2 700, 2 800)], with a total path length of 11, 000 mm, an elbow count of 2, and an energy area ration (the proportion of path length that traverses the energy area relative to the total path length) of 32.0%. In their respective 20 iterations, the ABC algorithm achieved the optimal layout in 5 times, whereas the IABC algorithm achieved it in 17 times. The IABC algorithm also showed a 37.4% reduction in the average fitness value and a 20.0% reduction in the average number of elbows compared with the ABC algorithm. Additionally, regarding the average energy area ratio, the IABC algorithm exhibited a 9.3% improvement over the ABC algorithm. These results indicate that the IABC algorithm can avoid local optima and has stronger global optimization capabilities than the ABC algorithm. Furthermore, the IABC algorithm demonstrates a faster convergence speed, with a 29.9% improvement compared with the ABC algorithm. Overall, the IABC algorithm outperforms the ABC algorithm in single pipeline routing optimization, offering enhanced solution quality and faster convergence speed.



1-Main engine; 2-Fuel tank 1; 3-Fuel tank 2; 4-Oil purifier 1; 5-Oil purifier 2; 6-Oil purifier supply pump; 7-Diesel daily tank; 8-Fuel oil daily tank; 9-Supply pump; 10-Self cleaning filter

Fig. 10 Layout space of fuel oil piping system in engine room

Table 3 Diagonal coordinates of some equipment in the engine room in the layout space

Equipment number	Equipment name	Equipment diagonal coordinate/mm
1	Main engine	(1 900, 5 200, 1 800) − (6 600, 6 800, 4 900)
2	Fuel tank 1	(9 800, 2 500, 0) − (13 000, 4 300, 1 000)
3	Fuel tank 2	(9 800, 7 700, 0) − (13, 000, 9 500, 1 000)
4	Oil purifier 1	(4 400, 1 000, 1 800) − (5 600, 2 700, 3 000)
5	Oil purifier 2	(6 000, 1 500, 1 800) − (7, 200, 2 700, 3 000)
6	Oil punfier supply pump	(11 500, 5 800, 1 400) − (12 100, 6 200, 2 000)
7	Diesel daily tank	(4 400, 9 800, 800) − (6 400, 11 000, 1 800)
8	Fuel oil daily tank	(7 200, 9 800, 800) − (9, 200 11 000, 1 800)
9	Supply pump	(6 500, 10 200, 2 400) − (7, 100, 10 600, 3 000)
10	Self cleaning filter	(3 400, 10 200, 3 800) − (4 200, 10 600, 4 500)

Table 4 Statistical data of single pipeline results calculated by ABC and IABC algorithms

Parameter	Algorithm	
	ABC	IABC
Optimal fitness value	0.243 1	0.243 1
Average fitness value	0.388 9	0.243 3
Fitness standard deviation	0.088 8	0.000 4
Average convergence generations	117.6	82.4
Average path length/mm	11 100	11 100
Average elbow nunber	2.5	2.0
Average energy area ratio/%	28.2	31.1

In this simulation experiment, the optimal solutions obtained by both the ABC and IABC algorithms were identical. After ranking the results from best to worst for each algorithm, the 10th result in each ranking was selected for comparison. The layout rendering of the algorithms is shown in Fig. 11, and the convergence curves are shown in Fig. 12.

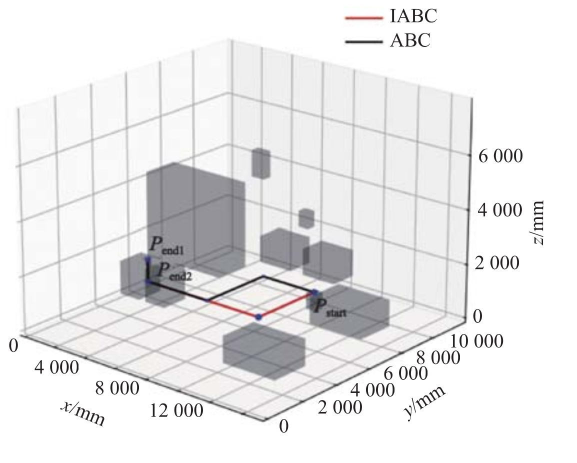


Fig. 11 Layout rendering of single pipeline calculated by ABC and IABC algorithms

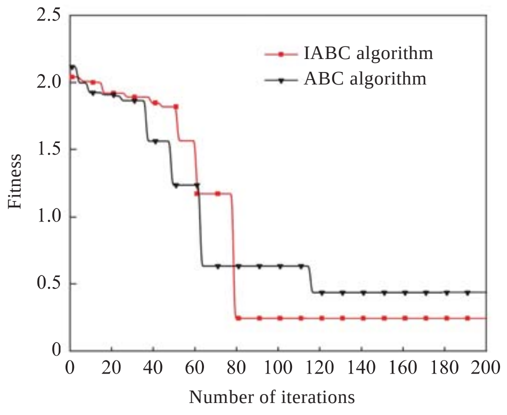


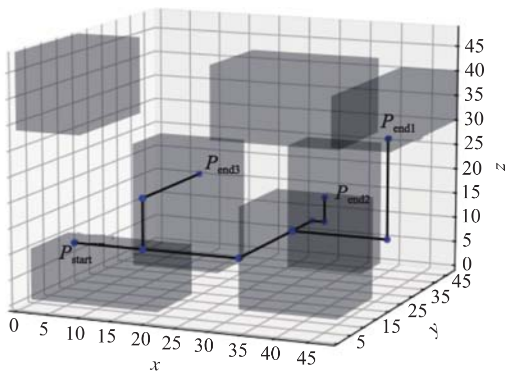
Fig. 12 The convergence curves of single pipeline calculated by ABC and IABC algorithms

3.2 Experimental verification and simulation example of branch pipeline routing optimization

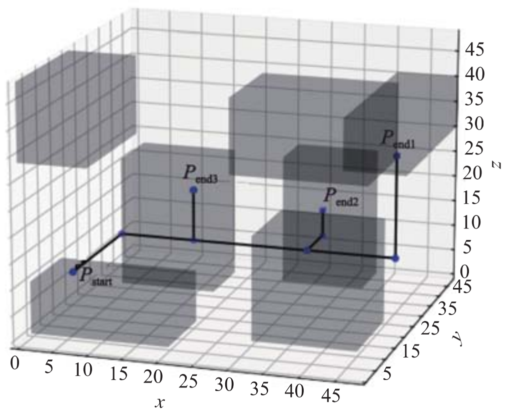
To verify the feasibility and efficiency of IABC algorithm for addressing branch pipeline issues, a comparison of calculation results by IABC-based coevolutionary algorithm and algorithm in Ref. [14] is shown in Fig. 13. Data comparison of branch pipeline calculation results is summarized in Table 5. The parameter settings for IABC algorithm are the same as those used for single pipeline calculations.

It can be seen from the results that both the algorithm in this paper and the one in Ref. [14] can obtain feasible solutions by connecting three branch pipelines through two T-joints. However, in terms of path length and the number of elbows, the results obtained by the algorithm in this paper are superior to those in Ref. [14].

To verify the feasibility and efficiency of the IABC-based coevolutionary algorithm proposed in this paper for the actual layout problem of ship branch pipelines, the engine room model in Section



(a) Calculation results of algorithm in Ref. [14]



(b) Calculation results of IABC-based coevolutionary algorithm

Fig. 13 Comparison of calculation results of branch pipeline based on experimental models in Ref. [14]

Table 5 Data comparison of branch pipeline calculation results by IABC-based coevolutionary algorithm and algorithm in Reference [14]

Pararneters	Algorithm in Ref. [14]	IABC-based coevolutionary algorithm	Reduction rate/%
Path length/mm	12 600	10 400	17.5
Number of elbows	5	3	40.0

3.1 is used. The pipeline from oil purifier 2 to the diesel daily tank and fuel oil daily tank is selected as simulation experiment object of the branch pipeline, with starting coordinates of (7 200, 2 100, 2 200), and the ending coordinates of (5 400, 9 800, 1 500) and (8 200, 9 800, 1 500). ABC and IABC algorithms are used for routing optimization, and the calculation was performed 20 times. The statistics of the calculation results are shown in Table 6. The optimal layout scheme obtained by the ABC algorithm is as follows: the main pipeline path is [(7 200, 2 100, 2 200), (7 200, 8 400, 2 200), (7 200, 8 400, 1 500), (7 200, 9 800, 1 500), (5 400, 9 800, 1 500)], while the branch pipeline path is [(7 200, 9 800, 1 500), (8 200, 9 800, 1 500)]. The total path length is 11 200 mm, with 3 elbows, and the energy area accounts for 29.0%. Using IABC algorithm, the optimal layout solution obtained is as follows: The main pipeline path is [(7 200, 2 100, 2 200), (7 200, 9 800, 2 200), (7 200, 9 800, 1 500), (5 400, 9 800, 1 500)], and the branch pipeline path is [(7 200, 9 800, 1 500), (8 200, 9 800, 1 500)]. The path length is 11,200 mm, with 2 elbows, and the energy area occupancy is 31.0%.

Table 6 Statistics of branch pipeline calculation results by ABC-based and IABC-based coevolutionary algorithms

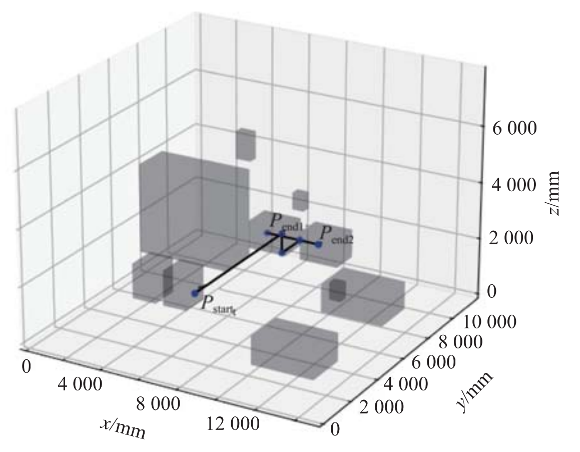
Parameter	Algorithm	
	ABC	IABC
Optimal fitness value	0.617 0	0.374 6
Average fitness value	0.713 7	0.483 2
Fitness standard deviation	0.145 7	0.095 1
Average convergence generations	124.6	102.5
Average path length/mm	11 600	11 550
Average elbow nunber	2.70	2.15
Average energy area ratio/%	26.0	28.4

In terms of average fitness value, average path length, and average elbow numbers, IABC algorithm showed reductions of 32.3%, 0.4%, and 20.4%, respectively, compared with ABC algorithm. IABC algorithm also increased the proportion of energy area by 9.2% compared with ABC algorithm. In addition, the convergence speed of the IABC algorithm has been improved by 17.7% compared to the ABC algorithm. In conclusion, when dealing with branch pipeline problems, the IABC-based coevolutionary algorithm has superior solutions and faster convergence speed than the ABC-based coevolutionary algorithm.

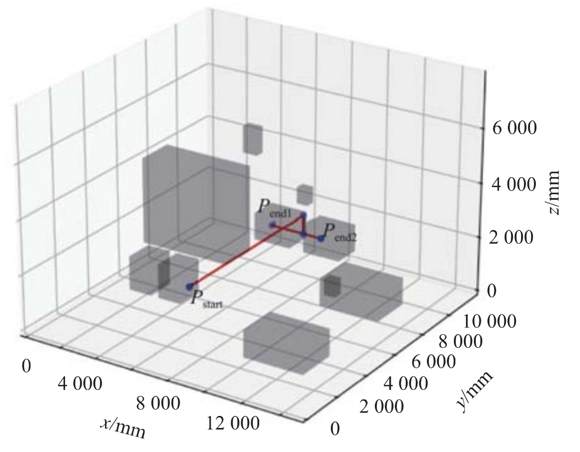
The optimal solutions obtained using two algorithms in the simulation experiments are selected, and the layout rendering is illustrated in Fig.14. The convergence curves are shown in Fig. 15.

4 Conclusions

This paper addresses the challenges of pipeline system layout in ship engine rooms by applying the ABC algorithm to pipeline routing optimization and using coevolutionary concepts to tackle the routing optimization of branch pipelines. Considering the susceptibility to local optima and low computational efficiency encountered when solving routing optimization problems with the ABC algorithm, an improved IABC algorithm is proposed. This algorithm incorporates crossover operations from genetic algorithms into the updating method for the following bees and adopts an adaptive strategy to enhance global optimization



(a) Layout of ABC-based coevolutionary algorithm



(b) Layout rendering of IABC-based coevolutionary algorithm

Fig. 14 Layout rendering of branch pipeline calculated by ABC-based and IABC-based coevolutionary algorithms

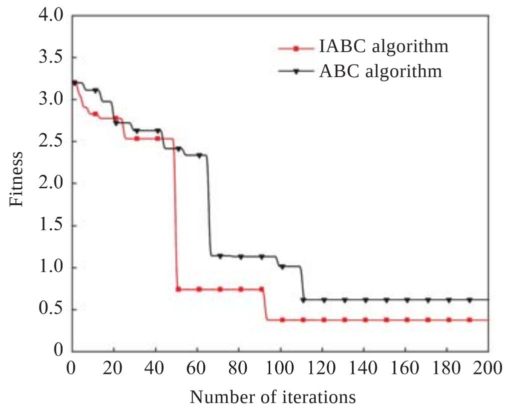


Fig. 15 The convergence curves of branch pipeline calculated by ABC-based and IABC-based coevolutionary algorithms

capabilities. Additionally, a more suitable path update method for the scout bees is introduced, which strengthens their role and value, leading to improved convergence speed. The simulation experiments are as follows:

1) In solving routing optimization problems of single pipeline, IABC algorithm improves the layout effect by 37.4% and convergence speed by 29.9% compared to ABC algorithm.

2) In solving routing optimization problems of branch pipeline, IABC algorithm improves layout effect by 32.3% and convergence speed by 17.7% compared with ABC algorithm.

IABC algorithm markedly outperforms ABC algorithm in both solutions quality and convergence speed. Compared with conventional pipeline layout methods, using IABC algorithm significantly saves design resources, enhances design efficiency, and promotes the development of intelligent and automated pipeline workflow in ship pipeline layout.

References

[1] KARABOGA D. An idea based on honey bee swarm for numerical optimization [R]. Kayseri, Turkey: Erciyes University, 2005.

[2] WANG H Q, HU Y Y, LIAO W D, et al. Path planning algorithm based on improved artificial bee colony algorithm [J]. Control Engineering of China, 2016, 23(9): 1407−1411 (in Chinese).

[3] ZHANG Y, GONG J, TANG Z Y, et al. Method for intelligent aeroengine pipeline layout based on improved multi-objective artificial bee colony algorithm [J]. Journal of Mechanical Engineering, 2022, 58(4): 277−284 (in Chinese).

[4] HE J R, HE G J, YU X S. The UAV path planning based on improved artificial bee colony algorithm [J]. Fire Control & Command Control, 2021, 46(10): 103−106 (in Chinese).

[5] LI Y S, WAN Y, ZHANG Y, et al. Path planning for warehouse robot based on the artificial bee colony-adaptive genetic algorithm [J]. Chinese Journal of Scientific Instrument, 2022, 43(4): 282−290 (in Chinese).

[6] LUO W P, LIU W Q, WANG H X, et al. Optimization of the section structure of container ship based on artificial bee colony algorithm and finite element strength calculation [J]. Chinese Journal of Ship Research, 2023, 18(2): 160−167, 217 (in Chinese).

[7] LIN Y, XIN D Y, BIAN X Y, et al. Improved particle swarm algorithm with adaptive inertia weight and its application in nuclear power pipeline layout [J]. Chinese Journal of Ship Research, 2023, 18(3): 1−13 (in both Chinese and English)

[8] LIN Y. An intelligent ship pipeline layout design software based on coevolutionary algorithm and swarm intelligence optimization algorithm: 2019SR0635885 [P]. 2022-12-06 (in Chinese)

[9] LIN Y. Software of ship branch pipeline automatic arrangement based on shortest path fast algorithm: 2020SR1056172 [P]. 2022-12-06 (in Chinese)

[10] LIN Y. Program software for automatic piping layout of nuclear power primary circuit system based on elite retention strategy genetic algorithm: 2022SR132010 [P]. 2022-12-06 (in Chinese)

[11] BIAN X Y, LIN Y, DONG Z R. Auto-routing methods for complex ship pipe route design [J]. Journal of Ship-

Production and Design, 2022, 38(2): 100 – 114.

[12] DONG Z R, BIAN X Y. Ship pipe route design using improved A* algorithm and genetic algorithm [J]. IEEEAccess, 2020, 8: 153273−153296.

[13] POLAT L, ACAN A, UNVEREN A, et al. Cooperative coevolutionary algorithms for fuzzy vehicular routing problem: An analysis of efficiency vs. geographical distribution [C]//2007 IEEE Congress on Evolutionary Computation.Singapore: IEEE, 2007.

[14] WU J. Coevolutionary optimization algorithm for ship branch pipe routing [D]. Dalian: Dalian University of Technology, 2008 (in Chinese).

[15] DONG Z R, LIN Y. A particle swarm optimization based approach for ship pipe route design [J]. International Shipbuilding Progress, 2017, 63(1/2): 59−84.

[16] JIANG W Y, LIN Y, CHEN M, et al. A co-evolutionary improved multi-ant colony optimization for ship multiple and branch pipe route design [J]. Ocean Engineering, 2015, 102: 63−70.

[17] FAN X N. A study of optimization methods for ship pipe routing design and application [D]. Dalian: Dalian University of Technology, 2006 (in Chinese).

基于改进人工蜂群算法的船舶管路路径寻优算法分析

李铁骊[1], 王文双[1], 刘海洋[2,3], 杨远松[2,3], 林焰[*1]

1 大连理工大学 船舶工程学院, 辽宁 大连 116024

2 中核绿色建造技术与装备重点实验室, 北京 101300

3 中国核工业二三建设有限公司, 北京 101300

摘　要: [**目的**] 人工蜂群(ABC)算法具有控制参数少、局部寻优能力强、收敛速度快的特点, 但在解决路径寻优问题方面, 存在容易陷入局部最优的缺陷。为解决船舶管路系统中的管路路径规划问题, 提出一种改进的人工蜂群(IABC)算法。[**方法**] 在传统人工蜂群算法的基础上, 在跟随蜂的更新机制中引入遗传算子中的交叉操作, 并对交叉算子的交叉概率采用自适应的策略; 通过对种群进行的交叉操作寻找全局范围内的新解, 并改进侦察蜂寻找新路径的方式, 由原来的对路径经过的点进行更新改为对路径中的"路段"进行更新; 随后, 提出一种适应于解决分支管路路径寻优的改进人工蜂群协同进化算法。[**结果**] 实例验证表明, 改进后的人工蜂群算法相比标准人工蜂群算法其路径布置效果能够提升 32.3%~ 37.4%, 收敛速度能够提升 17.7%~29.9%。[**结论**] 无论是解决单管路还是分支管路, 改进后的人工蜂群算法相比传统的人工蜂群算法求解质量更高、收敛速度更快、稳定性更好。

关键词: 船舶管路; 人工蜂群算法; 路径规划; 协同进化