frontiers | Frontiers in Industrial Engineering

TYPE Original Research
PUBLISHED 05 March 2025
DOI 10.3389/fieng.2025.1523203

OPEN ACCESS

EDITED BY
Mitsuo Gen,
Fuzzy Logic Systems Institute, Japan

REVIEWED BY
Wenqiang Zhang,
Henan University of Technology, China
Jie Gao,
Xi'an Jiaotong University, China
Kaphwan Kim,
Pusan National University, Republic of Korea

*CORRESPONDENCE
Dong Tang,
tangdong0112@stu.shmtu.edu.cn

RECEIVED 05 November 2024
ACCEPTED 06 January 2025
PUBLISHED 05 March 2025

CITATION
Liang C, Tang D, Zhao R and Wang Y (2025) Hybrid genetic algorithm and Q-learning-based solution for the time-variant berth and quay crane allocation problem.
Front. Ind. Eng. 3:1523203.
doi: 10.3389/fieng.2025.1523203

Hybrid genetic algorithm and Q-learning-based solution for the time-variant berth and quay crane allocation problem

Chengji Liang, Dong Tang*, Rui Zhao and Yu Wang

Institute of Logistics Science and Engineering of Shanghai Maritime University, Pudong, China

Introduction: This study addresses the joint scheduling optimization of continuous berths and quay cranes by proposing a time-variant quay crane allocation method.

Methods: A coordinated optimization model is constructed that considers the temporal dimension of quay crane scheduling and equipment collision factors to reduce overall port operational costs. A hybrid intelligent algorithm integrating Q-learning is innovatively designed, using a genetic algorithm as the main framework while embedding a quay crane allocation module and dynamically selecting genetic operators through Q-learning to achieve adaptive optimization of the evolutionary mechanism.

Results: The module with Q-learning optimization is compared to the module without Q-learning optimization, demonstrating that the Q-learning module can accelerate the convergence of the algorithm and has a better ability to find the optimal solution in large-scale cases, proving the effectiveness of the module.

Discussion: The results show that the proposed algorithm and CPLEX perform similarly in small-scale cases, while the solution speed and capability are better than the genetic algorithm in large-scale problems and superior to the CPLEX algorithm with time constraints in some cases, proving the effectiveness and superiority of the proposed algorithm.

KEYWORDS

continuous berth, integrated berth and quay crane allocation, time-variant quay crane allocation, Q-learning, genetic algorithm

1 Introduction

Maritime transportation accounts for 90% of global trade (Liang et al., 2011). As the key link between maritime trade and other modes of transportation, the operational efficiency of ports directly affects the speed and cost of trade flows (Liu, 2020). Berths and quay cranes are the main factors in port operation and management that enhance the core competitiveness of container ports.

In planning berthing schedules, terminal managers must solve the problem of allocating berths and quay cranes. The berth allocation problem (BAP) assigns ships to specific areas along the terminal, aiming to minimize both total service time and waiting time. This is essential for reducing vessel turnaround and increasing terminal throughput. After berths are allocated, the quay crane allocation problem (QCAP) follows, determining the number of quay cranes assigned to each ship, their location, and their working time. The goal of

QCAP is to minimize the time needed to unload and load containers while considering crane operational limits.

In the actual operation process, the berth problem and the quay crane allocation problem have a high correlation (Zheng et al., 2020). On the one hand, the berth problem determines the berthing position and berthing time of a ship, which directly affect the number of quay cranes allocated and the working time of quay cranes. On the other hand, the departure time of the ship depends on the number of allocated quay cranes and the working efficiency (Lu et al., 2011). Therefore, the simultaneous planning of berth allocation and quay crane allocation can improve the operational efficiency of the terminal and the utilization rate of berths. The quay crane allocation problem can be categorized into the static quay crane allocation problem and the time-variant quay crane allocation problem by distinguishing whether the quay cranes are moving during the unloading and loading process of the ship (Ng and Mak, 2006). Static quay crane assignment means that the quay cranes assigned to the same ship start working at the same time, and even if the quay cranes complete their tasks in advance, they must wait for the other quay cranes to complete their work before they can be released together. Time-variant quay crane assignment means that a quay crane can be moved to service another ship before the designated ship completes all unloading and loading tasks. Compared with the static allocation problem, the time-variant allocation can better utilize the resources of the quay crane and improve the utilization rate of the terminal resources.

This study addresses the joint problem of berth allocation and time-variant quay crane allocation, aiming to minimize total port costs, which include vessel costs, crane movement costs, and service costs. Traditional metaheuristic algorithms often struggle with slow convergence and poor optimization, as they fail to adapt solution generation based on the current state. To resolve this, a Q-learning-based genetic algorithm is proposed. Q-learning is used to evaluate the current population state and guide the generation of new solutions, improving optimization performance. Specifically, a global search algorithm is proposed to solve the time-variant quay crane allocation problem by considering both quay crane allocation and berth allocation to improve the utilization of port resources. We store the quay crane allocation as a code in the chromosome, which is taken into consideration at the berth stage, and use the genetic algorithm to perform a global search and the quay crane allocation algorithm to perform the quay crane allocation. In order to improve the search ability of the algorithm, the Q-learning algorithm is introduced to select the appropriate genetic operation from ten genetic operators according to the current state of the population, and the feedback obtained from the execution results is passed into the Q-learning algorithm as a reward to train it and improve the search ability of the solution. It effectively overcomes the problem of high randomness and the tendency to fall into local optimization that plagues traditional genetic algorithms. The scheme in this article can achieve high utilization of berth and quay crane resources and also provide strong technical support for port operation optimization.

The remainder of this article is organized as follows. Section 2 is the literature review. Section 3 provides a detailed description of the research problem and model. Section 4 presents the algorithms used in this article. Section 5 discusses the experimental results. Section 6 offers conclusions and future directions.

2 Literature review

In this section, we summarize the current state of research on the joint allocation of berths and quay cranes and the methods for their solution.

Park and Kim (2003) first proposed the problem of joint allocation of berths and quay cranes. They constructed an integer programming model and proposed a two-stage approach to solve the problem: the first stage determines the berth location, berthing time, and the number of required quay cranes for a ship; the second stage assigns specific quay cranes to the ship to provide services. Meisel and Bierwirth (2009) discussed the problem of continuous berthing and quay crane allocation. Based on this, they discussed the problem of reduced efficiency of individual quay cranes due to mutual influence on each other during the joint operation of multiple quay cranes. They used a mixed integer linear programming (MILP) method and two metaheuristic algorithms for the solution. The above scholars consider the berth problem and the quay crane allocation problem in two steps; that is, only the number of quay cranes required by the ship is decided when the berth is allocated, and the allocation of specific quay cranes is considered after the ship enters the berth. Türkoğulları et al. (2014) considered the quay cranes along with the berth allocation, and the quay crane resources were considered during the berth planning. They developed a cutting plane algorithm to solve the problem and achieved a joint berth and quay crane allocation by iteratively solving the berth allocation and quay crane assignment (number) problems (BACAP) with additional constraints added. Correcher et al. (2019) built on this foundation by proposing a new quay crane allocation model and solving the problem using a branch-and-bound approach. Most current studies focus on the static quay crane allocation problem, and there are few studies on the time-variant quay crane allocation problem. Chang et al. (2010) proposed a rolling time-domain strategy considering the working time of variable quay cranes and constructed a feasible solution heuristic algorithm and a parallel genetic algorithm to solve the time-variant quay crane allocation problem. Krimi et al. (2020) proposed a mathematical model for continuous berth and time-variant quay crane allocation considering realistic constraints. They evaluated the feasibility of the model using CPLEX. However, considering the issue of solution efficiency, they designed heuristics for a general variable neighborhood search to address the problem. The results indicate that the designed algorithm meets the need for providing high-quality solutions in a short period of time. Malekahmadi et al. (2020) applied a particle swarm algorithm to solve the time-variant quay crane allocation problem by considering tidal factors for comparison.

Most scholars currently use metaheuristic algorithms to solve the problem of joint allocation of berths and quay cranes. For the static quay crane allocation problem, Ji et al. (2022) solved the static joint berth and quay crane allocation problem by means of a rolling horizon program and the ALNS algorithm embedded in a time-variant consideration of unplanned vessel entries. Correcher and Alvarez-Valdes (2017) proposed a metaheuristic approach to solve the static BACASP problem by means of a biased stochastic key genetic algorithm with simulated features and multiple local search processes.

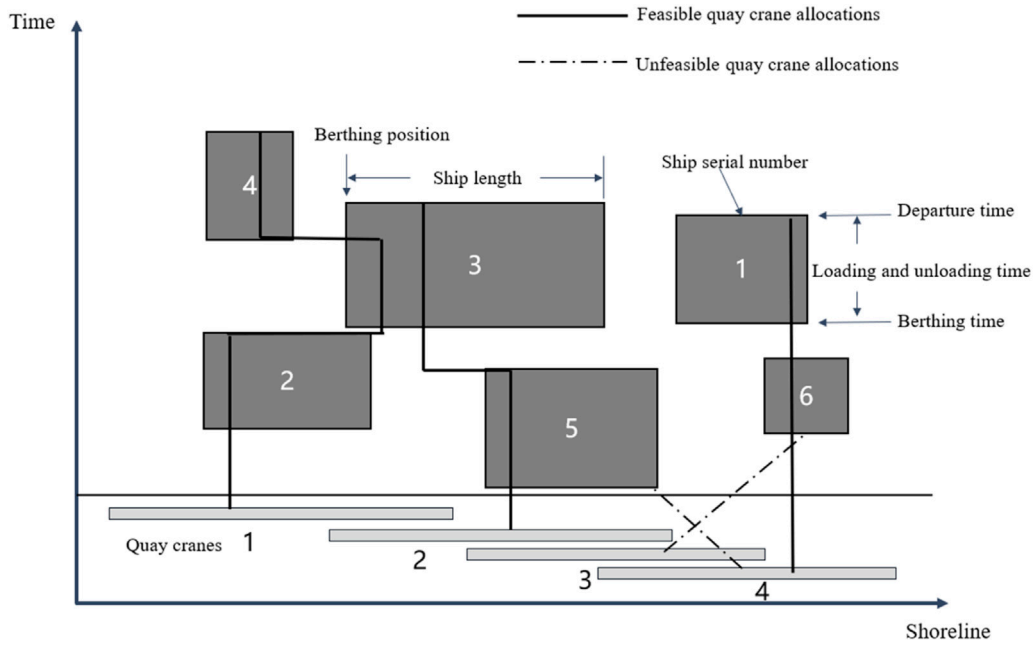

FIGURE 1
Illustrative diagram of time-variant quay crane allocation.

There are two main solution methods for the time-variant quay crane allocation problem in existing studies: 1) Calling the mixed integer programming (MIP) model for rolling solution within a certain time window. 2) A step-by-step solution using the MIP method to solve the problem within the time window. For example, Agra and Oliveira (2018) proposed an integer linear programming model and a rolling time window solution method for the time-variant quay cranes and berth joint scheduling problem, which is solved by using the MIP model within the time window. Karam and Eltawil (2016) decomposed the time-variant quay crane allocation problem into the berth allocation problem and the quay crane allocation problem, which are solved separately and integrated through the ship's demand for the number of quay cranes in a feedback loop. Thanos et al. (2021) divided the time-variant quay crane solution into three steps: first, determining the berthing position of the ship, then arranging the quay cranes within the range to serve the ship according to the berthing position of the ship, and finally deciding the time interval that the quay cranes serve the ship.

Through the above, it is not difficult to observe that there is no suitable joint allocation method to solve the global search solution in existing research. Instead, a step-by-step solution is used to address the problem, that is, determining the location of the quay cranes and then determining which quay cranes will provide the service.

3 Problem description and modeling

3.1 Problem description and assumption

Before a ship berths at the destination port, it needs to report the ship type, expected arrival time, amount of cargo to be loaded and unloaded, and expected departure time. Based on this information, the port assigns berths and quay cranes to ships to minimize the total port cost. The berth-quay crane allocation process can be mapped to a two-dimensional space-time diagram. As shown in Figure 1, the time-variant quay crane allocation allows the quay crane to leave to serve other vessels at any time during the execution of loading and unloading activities but not to cross other quay cranes while working. The solid line in the figure indicates a feasible allocation of quay cranes, and the dashed line indicates an infeasible allocation of quay cranes. The core of time-variant quay crane assignment is its flexibility, which allows the quay cranes to switch service between different vessels while ensuring safety and efficiency. For example, the No. 1 quay crane can provide service for the No. 4 vessel when the No. 3 vessel has not yet left the harbor; the No. 2 quay crane can connect to the unloading and loading operation of the No. 2 vessel after the completing its tasks at the No. 5 vessel. Such a time-variant scheduling program enables more flexible scheduling of quay crane resources and improves the loading and unloading efficiency of the terminal.

To achieve time-variant scheduling of quay cranes, we must determine which quay cranes serve each ship in each time period. We construct an integer programming model with the objective of minimizing the total cost of ship delay cost, quay crane service cost, and quay crane movement cost by considering the quay crane working range and constraints on quay crane crossing. In the model, we take the ship berthing time, berthing position, and the serial number of the ship served by each time period of the quay crane as decision variables. In order to better study this problem, the following assumptions are made in the next discussion: 1) Ship berthing is not limited by tide, water depth, mechanical failure, etc., 2) Ship unloading and loading services are continuous and cannot be stopped in the middle, and each ship has a range of the number of

TABLE 1 Sets, parameters, and decision variables of the proposed problem.

Set	Connotation
S	Set of ships arriving during the planning period. $S = \{1, 2...s\}$, indexed by i, j
T	Set of planning periods. $T = \{1, 2...t\}$
Q	Set of quay cranes in front of the terminal. $Q = \{1, 2...q\}$
Parameter	**Connotation**
C	Total cost of the berth allocation plan
$c_t^{service}$	Average cost of service per unit of time for each quay crane
c_i^{delay}	Average cost of compensation per unit of time for ship delays
c^{move}	The average cost of moving a quay crane
e_i	Time when ship i finishes unloading and loading cargo
d_i	Expected departure time of ship i
a_i	Expected arrival time of ship i
f_i	Time required for ship i to load and unload cargo
l_i	Length of ship i
T^{max}	End of the planning period
Q^{max}	Total number of quay cranes
q_i^{max}	Maximum number of quay cranes that can be assigned to a ship
q_i^{min}	Minimum number of quay cranes that can be assigned to a ship
η_{it}	Average handling efficiency of ship i in time period t
w_i	Amount of cargo to be loaded and unloaded by ship i
q_k^l	Leftmost position that can be served by quay crane k
q_k^r	Rightmost position that can be served by quay crane k
r	Safe distance between ships at berth
M	Infinite number
L	Length of the quay shoreline
Decision variables	**Connotation**
z_{iq}^t	Binary variable that takes the value of 1 if ship i is served by quay crane q in time slot t and 0 otherwise
v_{iq}^t	Binary variable that takes the value of 1 if quay crane q moves to ship i in time period t and 0 otherwise
b_i	Berthing time of ship i
p_i	Berthing position of ship i
y_{ij}	Binary variable that takes the value of 1 if the berthing position of ship i is to the left of ship j and 0 otherwise
x_{ij}	Binary variable that takes the value of 1 if the berthing time of ship i is earlier than that of ship j and 0 otherwise
m_{it}	Binary variable that takes the value of 1 if vessel i is serviced in time period t and 0 otherwise

quay cranes that can be allocated; 3) Quay cranes cannot cross each other; 4) The movement time of the quay cranes is negligible.

3.2 Mathematical formulation

For better understanding, we provide an explanation of the notation used in the model in Table 1.

We construct an integer programming model with the objective of minimizing the total ship delay cost, quay crane service costs, and quay crane movement costs by taking into account the quay crane working range, constraint limitations of quay crane crossings, and other elements as follows:

$$\min\ C = \sum_{i\in S}\left(\sum_{t\in T} c_t^{\text{service}} \sum_{q\in Q} z_{iq}^t + c_i^{\text{delay}}(e_i - d_i) + c^{\text{move}} \sum_{t\in T}\sum_{q\in Q} v_{iq}^t\right). \tag{1}$$

Equation 1 addresses the objective function, that the total cost of the port is the lowest, and has three components: the first term is the cost of the quay cranes service, the length of time the quay cranes service the ship, and the cost of the quay cranes at different moments of the service. The second is the cost of the ship's delays; this item will only be established if the ship's departure time is later than the expected time of departure; otherwise, it will be zero. The third is the movement cost of the quay cranes, which comes from the personnel dispatch and resource consumption caused by the stopping and restarting of the quay cranes and is established when the quay cranes change the service ship or serve the ship for the first time.

$$\text{s.t.} \quad b_i \geq a_i, \quad \forall i \in S. \tag{2}$$

Equation 2 ensures that each ship berths later than its arrival time, avoiding impractical scheduling arrangements.

$$e_i = b_i + f_i, \quad \forall i \in S. \tag{3}$$

Equation 3 calculates the departure time of a ship based on the loading and unloading time of the ship.

$$p_i + l_i + r \leq p_j + M \cdot (1 - y_{ij}), \quad \forall i, j \in S, \tag{4}$$

$$y_{ij} + y_{ji} \leq 1, \quad \forall i, j \in S, \tag{5}$$

$$e_i \leq b_j + M \cdot (1 - x_{ij}), \quad \forall i, j \in S, \tag{6}$$

$$x_{ij} + x_{ji} \leq 1, \quad \forall i, j \in S, \tag{7}$$

$$x_{ij} + x_{ji} + y_{ij} + y_{ji} \geq 1, \quad \forall i, j \in S. \tag{8}$$

A series of constraints from Equations 4–8 ensures that only one vessel can berth at the same location at any given time, avoiding scheduling conflicts in time or space.

$$p_i + l_i \leq L, \quad \forall i \in S. \tag{9}$$

Equation 9 requires that the berthing position of the ship be on the shoreline of the port and that the feasibility of the berthing position be ensured.

$$e_i \leq T^{\max}, \quad \forall i \in S. \tag{10}$$

Equation 10 requires all ships to leave port during the berth planning period.

$$\sum_i \sum_q z_{iq}^t \le Q^{\max}, \forall i \in S, \forall t \in T. \tag{11}$$

Equation 11 ensures that the total number of quay cranes serving ships at any one time is less than the total number owned by the terminal.

$$\sum_q z_{iq}^t \ge q_i^{\min}, \forall i \in S, \forall t \in T, \forall q \in Q, \tag{12}$$

$$\sum_q z_{iq}^t \le q_i^{\max}, \forall i \in S, \forall t \in T. \tag{13}$$

Equations 12, 13 require that the number of quay cranes on the service ship at each point in time be greater than the minimum required number of quay cranes and less than the maximum number of quay cranes that the ship can accommodate.

$$\sum_t \sum_q \eta_{it} z_{iq}^t \ge w_i, \forall i \in S, \forall t \in T. \tag{14}$$

Equation 14 ensures that the loading and unloading tasks of the ship can be accomplished.

$$M \cdot m_{it} \ge \sum_q z_{iq}^t, \forall i \in S, \forall t \in T, \tag{15}$$

$$m_{it} \le \sum_q z_{iq}^t, \forall i \in S, \forall t \in T. \tag{16}$$

Equations 15, 16 indicate that if ship i is serviced in time slot t, then m_{it} takes the value of 1 and otherwise 0.

$$f_i = \sum_t m_{it}, \forall i \in S. \tag{17}$$

Equation 17 calculates the total loading and unloading time for each ship.

$$\sum_{t=1}^{t_{\max}-1} |m_{i,t} - m_{i,t+1}| \le 2, \quad \forall i \in N. \tag{18}$$

Equation 18 ensures that the quay crane is available to serve the ship for each time period after it berths until the ship departs.

$$z_{ia}^t q_a^l - z_{jb}^t q_b^l + \left(z_{ia}^t + z_{jb}^t - 2\right)M \le \left(1 - y_{ij}\right)M, \forall i \in S, \forall j \in S, \forall t \in T, \forall a \in Q, \forall b \in Q. \tag{19}$$

Equation 19 ensures that the quay cranes do not cross. If ship i is to the right of ship j and y_{ij} is 0, the equation is constant; if ship i is to the left of ship j, the right-hand side of the equation is 0; and if both ships require quay crane service at the same time, the quay crane that serves ship i must be all the way to the left of the quay crane that serves ship j. The equation also ensures that the quay crane that serves ship j will not cross.

$$q_k^l - p_i - l_i \le M\left(1 - z_{ik}^t\right), \forall i \in S, \forall t \in T, \forall k \in Q, \tag{20}$$

$$p_i - q_k^r \le M\left(1 - z_{ik}^t\right), \forall i \in S, \forall t \in T, \forall k \in Q. \tag{21}$$

Equations 20, 21 provide that a ship can only be serviced within the working limits of the quay crane.

$$v_{iq}^t \le z_{iq}^t, \forall i \in S, \forall t \in T, \forall q \in Q, \tag{22}$$

$$v_{iq}^t \le 1 - z_{iq}^{t-1}, \forall i \in S, \forall t \in T, \forall q \in Q, \tag{23}$$

$$v_{iq}^t \ge z_{iq}^t - z_{iq}^{t-1}, \forall i \in S, \forall t \in T, \forall q \in Q. \tag{24}$$

Equation 22 through Equation 24 ensure that v_{iq}^t records the first time at time t that a quay crane q moves to a ship i to service it. Equation 22 ensures that the quay crane only moves to a ship if it must service that ship. Equation 23 states that when the quay crane q has already serviced ship i at the previous time period $t-1$, the move is not recorded. Equation 24 indicates that when quay crane q did not serve ship i in the previous time period $t-1$, and quay crane q serves ship i in this time period t, then a movement of the quay crane is recorded.

$$p_i \ge 0, b_i \ge 0, \quad \forall i \in S, \tag{25}$$

$$x_{ij}, y_{ij} \in 0, 1, \tag{26}$$

$$m_{it} \in 0, 1, \forall i \in S, \forall t \in T, \tag{27}$$

$$z_{iq}^t \in 0, 1, \forall i \in S, \forall t \in T, \forall q \in Q. \tag{28}$$

Equation 25 defines the non-negativity of berthing position and berthing time. Equations 26–28 define the 0–1 variables.

4 Solution methodology

Berth-quay crane scheduling is an NP-hard problem, and exact algorithms are difficult to solve in large-scale arithmetic cases (Lujan et al., 2021). Compared with traditional heuristic algorithms, genetic algorithms have higher global search capability (Hanagandi and Nikolaou, 1998). However, traditional genetic algorithms have a single genetic operation that easily falls into local optimization. To balance the diversity of the population and global search ability, we adopt the Q-learning algorithm to adaptively select the genetic operator (Wang et al., 2013). In the calculation process, we use the genetic algorithm to determine the berthing order, berthing position, and the number of quay cranes required for each time period. The quay crane allocation algorithm calculates the specific quay crane allocation scheme for each ship and evaluates the fitness of the solution. The Q-learning algorithm selects the genetic operator according to the state of the population in the genetic algorithm, which advances the iterative process of the genetic algorithm. The three algorithms continuously interact with each other and ultimately obtain the optimal solution that meets the requirements. The algorithm framework is shown in Figure 2.

4.1 Genetic algorithm

4.1.1 Chromosome and population initialization

We encode the solution into a three-layer structure: the first layer shows the order in which the ships berth, the second layer indicates the serial number of the first docking cluster pile where the ship docks, that is, where the ship berths, and the third layer is a number obtained from the list encoding, which contains the number of quay cranes needed in each time period after the ship berths. The structure of the solution is shown in Figure 3, taking Ship 1 as an example. Ship 1 is the third in the berthing order, the berthing cluster pile serial number is 44, and the number of quay cranes

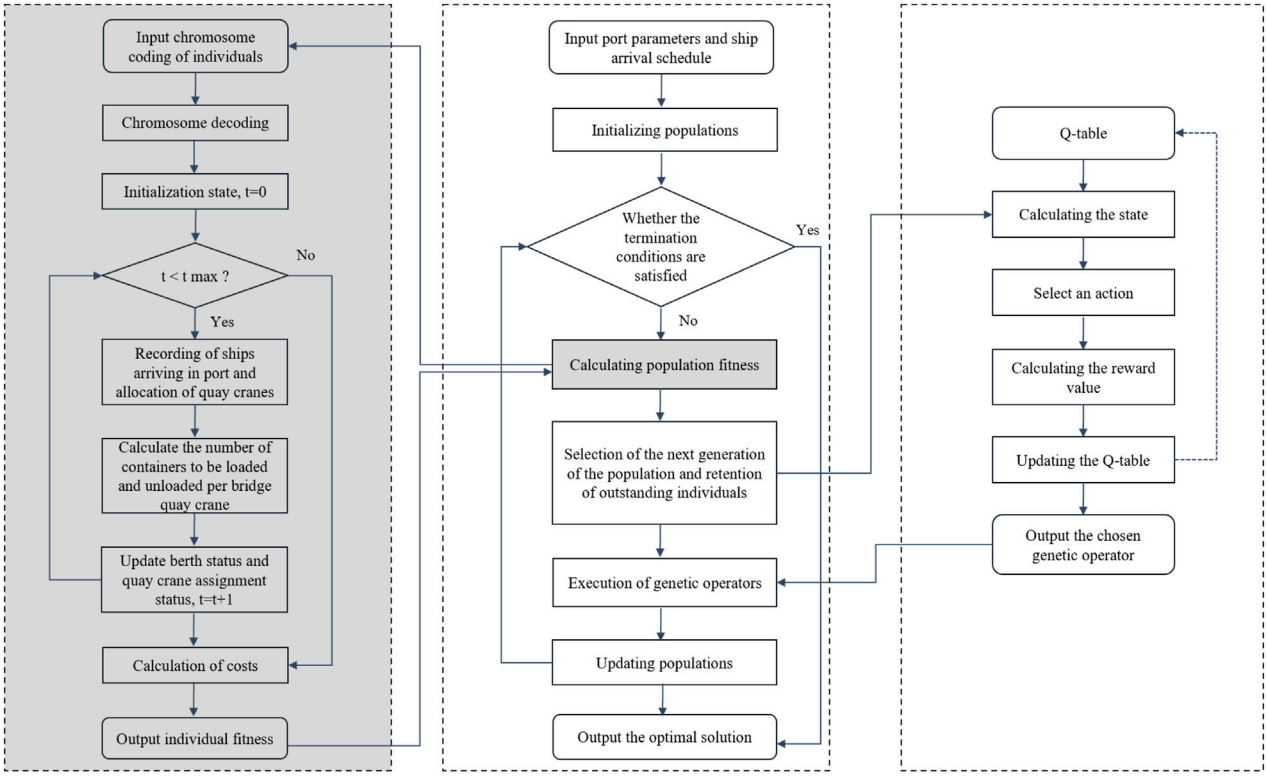

FIGURE 2
Algorithm framework diagram.

Ship number	1	2	3	4	5
Order of berthing	3	2	1	4	5
Position of berthing	44	32	45	78	96
Number of quay cranes required at each moment of berthing (code)	345157	21854	26462	85248	74382

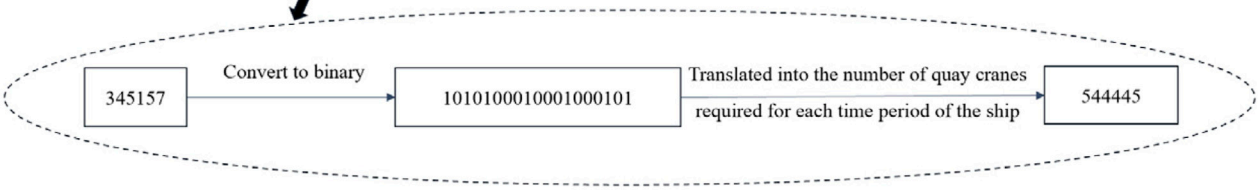

FIGURE 3
Structure of the solution.

required in the first time period after berthing is 5. The number of quay cranes required in the second time period is 4, and so on.

The third layer of the decoding consists of an array of an indeterminate number of quay cranes, which must be decoded to obtain the number of quay cranes required for each time period of the vessel after berthing. The logic of decoding is as follows: First, the encoded data are converted into binary numbers. Then, according to the number of quay cranes activated in the port, the number of binary bits corresponding to the number of quay cranes is obtained. Finally, according to the number of bits, it is decoded into the required number of quay cranes. Taking the third layer of encoding of Ship 1 as an example, the encoded number is 345,157, which is first converted into a binary number for storage. Assuming the port has eight available quay cranes, we convert the encoded number 345157 into a binary number (10101000100010001000010101). Reading four bits from back to front, we get 0101, which converts to the decimal number 5. By similar reasoning, we obtain the array: 54445, which means that over five time periods,

the ship requires 5, 4, 4, 4, and 5 quay cranes, respectively. Then, using the quay crane allocation algorithm, we allocate the specific serial numbers of the quay cranes.

4.1.2 Fitness calculation and quay crane allocation algorithm

4.1.2.1 Fitness calculation

We process individual chromosomes through a set of heuristic algorithms to compute the various costs of the port and to perform quay crane allocation. The berthing order of ships is first determined by the first layer of coding of the chromosome. Then, the second layer of the chromosome's coding is read to determine the berthing position. If the berthing position of the ship does not conflict with the berthing position of the previous ship, satisfies the safety distance constraint, and the quay cranes of the port can satisfy the minimum quay crane requirement of the ship, then the ship enters the port. Otherwise, the ship needs to wait for the previous ship to leave the port before entering the port. After the ship enters the port, the quay crane allocation algorithm is invoked to allocate a specific quay crane for the ship.When the unloading and loading of the ship's cargo is completed, the ship leaves the port, and the delay cost of all ships, the service cost of the quay cranes, and the movement cost of the quay cranes are calculated. The steps of the algorithm are as follows:

Step 1: input a chromosome, let $t = 0$, read the first layer of the chromosome to obtain the berthing order of the ships $S(k)$, let $k = 1$, set the port state to empty, the quay crane service state to empty, and the expected number of ships to be accepted is N.

Step 2: If $t > t^{\max}$ or $k > N$, and the port state is empty, then jump to Step 7; otherwise, jump to Step 3.

Step 3: If the arrival time of ship $S(k)$ is greater than t, the berthing position of ship $S(k)$ is not in conflict with the berthing position of the ship in the port, and if it satisfies the safety distance constraint, and if the quay cranes in the port are able to meet the minimum demand of the ship, then record the berthing position of ship $S(k)$, so that $k = k + 1$.

Step 4: Input the original quay crane service status and the expected number of quay cranes for ships in port. Activate the quay crane allocation algorithm to minimize the changes and update the service status, considering the working range and continuous allocation of quay cranes.

Step 5: Calculate the total amount of cargo to be loaded and unloaded by the ships in port at this moment based on the updated service status of the quay cranes and the efficiency of the quay cranes at time t, and update the total amount of cargo to be loaded and unloaded by the ships in port. Record the service status of the quay cranes and the movement of the quay cranes.

Step 6: Determine if any ship's unloading and tasks are completed. If so, update the port status, record the ship's departure time, and the quay cranes' service and movement status during the unloading and loading processes. Return to Step 2.

Step 7: If $k > N$, calculate the delay, service, and movement costs for the quay cranes based on each ship's departure time, the quay cranes' service, and movement situations. If not, mark the solution as infeasible and set the costs to Inf.

Step 8: Output the delay cost of the ship, the service cost of the quay cranes, and the movement cost of the quay cranes for this chromosome.

We set the fitness of the genetic algorithm to depend on the total cost of the port, and the fitness is defined by Equation 29, where x represents the individual, C_x^{delay} represents the delay cost of the ship, $C_x^{service}$ represents the service cost of the quay crane, and C_x^{move} is denoted as the movement cost of the quay crane.

$$\text{fitness}(x) = \frac{1000}{C_x^{\text{delay}} + C_x^{\text{service}} + C_x^{\text{move}}}. \tag{29}$$

4.1.2.2 Quay crane allocation algorithm

We design a quay crane allocation algorithm to solve for ship-specific assigned quay crane numbers. The number of quay cranes required for each time period after berthing of the ship is recorded in the chromosome, and the quay crane allocation algorithm turns these requirements into assigned quay crane numbers. The steps are as follows:

Step 1: Input the current port status $p(t)$, the last time period port status $p(t-1)$, and the last time period quay crane status $Q(t-1)$. Judge whether the state of the port in the last time period $p(t-1)$ and the state of the port in the current period are consistent. If they are consistent, it means that there is no new ship in the port, so proceed to Step 5; otherwise, go to Step 2.

Step 2: Identify the departing ship, mark the quay cranes that served it in the last period as unserved in the quay crane status $Q(t-1)$, and update the status to $Q(t-1) \leftarrow Q'(t-1)$. If there is no departing ship, no further processing is needed. Judge whether there is a new ship; if so, go to Step 3; otherwise, go to Step 5.

Step 3: Compare the last time period port state $p(t-1)$ with the current port state $p(t)$, find the new ship number k, and query the set of quay crane numbers that can be invoked at the berthing location $q(k)$.

Step 4: Retrieve the available quay crane number set $q(k)$ of ship k. Arrange the quay cranes for ship k without affecting the quay crane state $Q(t-1)$ in the previous time period, set the quay crane state at this time to $Q'(t-1)$, and make the state $Q(t-1) \leftarrow Q'(t-1)$.

Step 5: Review the current port state, noting the expected number of shore bridges (n_k^t) and the set of available quay crane numbers of ships $q(k)$ during the berthing time of the ships in port. Examine the quay crane state $Q(t-1)$ of the previous time period, marking the quay cranes assigned to ships as -1 and those unassigned as -2.

Step 6: Select quay cranes from the set of available quay cranes $q(k)$ for each ship k to fulfill the minimum number of quay cranes required for each ship. Prioritize the selection of quay cranes marked with -1 to ensure that their spacing

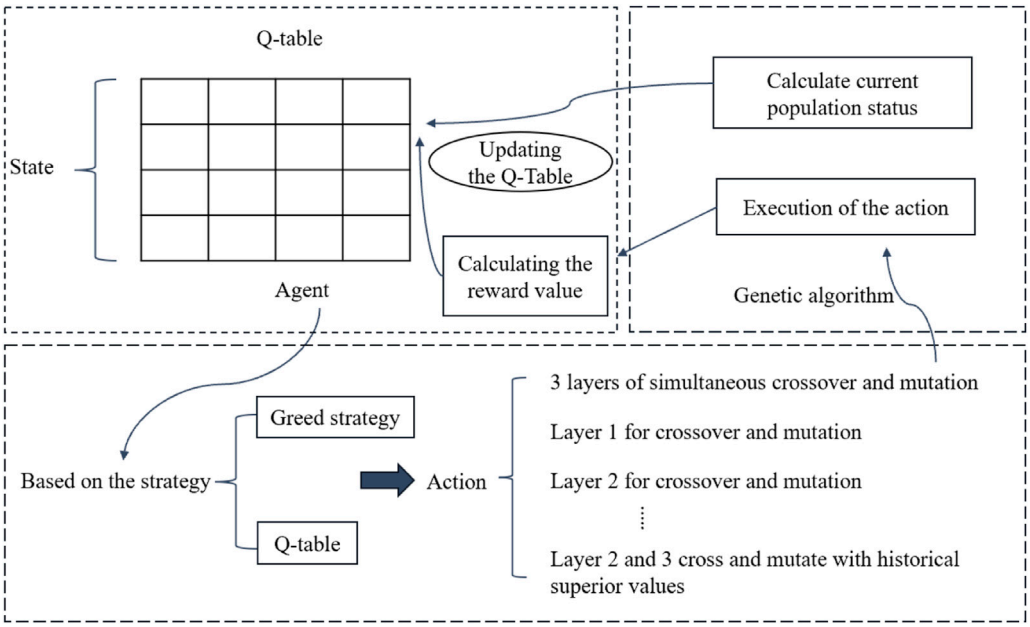

FIGURE 4
Q-learning algorithm framework.

aligns with the expected number required. Subsequently, select quay cranes that offer the largest spacing. Label quay cranes that have been allocated.

Step 7: Traverse through the ships in the port and prioritize the allocation of the quay cranes marked with −1 to the ships so that the change in the quay crane allocation is less than the previous quay crane state $Q(t-1)$. Mark the quay cranes that have been assigned.

Step 8: Analyze the port state, considering the future demand for quay cranes from ships. Assign quay cranes marked as −2 to ships, prioritize the ships with smaller changes in the expected number of quay cranes $(n_k^t, n_k^{t+1}, n_k^{t+2}...)$, and mark the completed allocation of the quay cranes.

Step 9: Record the quay crane allocation and output the quay crane status $Q(t)$.

4.1.3 Genetic operations

We use roulette to select the operator and maintain the elite strategy (Pham et al., 2024). The three-layer encoding approach involves three distinct crossover and mutation operations. The crossover operations for the three chromosome layers include partial matching crossover, single-point crossover, and single-point crossover applied after decoding. Mutation operations for the three chromosome layers consist of exchange mutation, random mutation, and bit-flip mutation. Instead of simultaneously applying crossover and mutation to all three chromosome layers, we leverage a pre-trained Q-learning algorithm to select appropriate genetic operators for one or more layers. This approach enhances the algorithm's convergence speed and optimization capability.

Two types of infeasible solutions can occur when performing genetic operations. We need to fix the infeasible solutions.

1) The new individuals may not satisfy the condition where the ship's berthing position, combined with its length, exceeds the total length of the quay shoreline. We employ random regeneration of berthing positions to ensure the ship's stern does not extend beyond the quay shoreline's total length.
2) The third layer of coding in the new individual may not meet the requirements for ship loading and unloading. We randomly add quay cranes to the list after decoding the third layer until the number and efficiency of quay cranes meet the ship's loading and unloading needs.

4.2 Q-learning algorithm

We utilize the Q-learning algorithm to determine the genetic operators during the population iteration process and set the agent of the Q-learning as populations. The two-dimensional states and actions form a three-dimensional Q-table. The agent calculates the current state of the population and selects the appropriate action (i.e., genetic operator) to enhance the genetic algorithm's convergence and optimization capabilities. The algorithm framework of the Q-learning part is shown in Figure 4.

We define the state of the environment in which the agent is located to be determined by the diversity of individuals in the population and the number of repetitions of the optimal individual fitness in the population that have not been updated. The diversity of individuals in the population is defined by the information entropy (as shown in Equation 30), where x_i is the individual fitness in the population, $p(x_i)$ is the probability of occurrence of the individual fitness, $-\sum_{i=1}^{n} P(x_i)\log_2 P(x_i)$ is the information entropy of the population, which is $\log_2 n$ when the population is completely disordered, and $H(x)$ is defined as the complexity of the population. The number of repetitions without

TABLE 2 Q-learning state definitions.

State	Details
State 1	$0 \le H(X) < 0.25, iter < 10\% \cdot G$
State 2	$0.25 \le H(X) < 0.5, iter < 10\% \cdot G$
State 3	$0.5 \le H(X) < 0.75, iter < 10\% \cdot G$
State 4	$0.75 \le H(X) < 1, iter < 10\% \cdot G$
State 5	$0 \le H(X) < 0.25, iter \ge 10\% \cdot G$
State 6	$0.25 \le H(X) < 0.5, iter \ge 10\% \cdot G$
State 7	$0.5 \le H(X) < 0.75, iter \ge 10\% \cdot G$
State 8	$0.75 \le H(X) < 1, iter \ge 10\% \cdot G$

updating represents the number of times the current individual repeats the optimal state keeping.

$$H(X) = \frac{-\sum_{i=1}^{n} P(x_i)\log_2 P(x_i)}{\log_2 n}. \tag{30}$$

We define the state space as in Table 2, where *iter* is the number of repeated non-updates, and G is the total number of iterations.

Traditional genetic algorithms often employ a single crossover and mutation method, which can easily lead to local optima. Moreover, these operations are applied to the entire chromosome, making local adjustments challenging for relatively fit individuals. We implement diverse crossover and mutation strategies at three coding levels: ship berthing order, ship berthing position, and the number of quay cranes required for ships during different time periods. The actions of the Q-learning algorithm in this study are defined in Table 3.

We set two strategies in selecting the action: the first one is to use the Q-table for selection, and the other one is to select by greedy strategy. The greedy value is defined in Equation 31, where $\varepsilon_{\max}$ is the hyperparameter represents the maximum greedy rate, f is the current iteration number, and G is the total iteration number. We use random selection of actions when $random < \varepsilon$; otherwise, one is randomly selected in the Q-table from the actions corresponding to the first three Q-values of the current state.

$$\varepsilon = \frac{\varepsilon_{\max}}{1 + e^{10\times\left(\frac{f-0.6\times G}{G}\right)}}. \tag{31}$$

After performing an action, the agent obtains a reward for that action, and we set the reward value as in Equation 32. The Q-table is updated as shown in Equation 33, where $Q(s_t, a_t)$ denotes the Q-value for each generation of selecting action a_t based on the state s_t. $\alpha \in [0, 1]$, and $\gamma \in [0, 1]$ denote the learning rate and discount factor. $\max Q(s_{t+1}, a_{t+1})$ denotes the maximum Q-value at the next state s_{t+1} when taking the next action a_{t+1}.

$$reward = \begin{cases} +1 & \text{if } f_{\text{gbest}}^{\text{new}} - f_{\text{gbest}}^{\text{old}} > 0, \\ 0 & \text{otherwise.} \end{cases} \tag{32}$$

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha\left(\text{reward}_{t+1} + \beta \max Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\right). \tag{33}$$

TABLE 3 Q-learning action definitions.

Scope	Primitive operator
Layer 1 encoding	(1) Two nodes are randomly selected, and all elements within the two nodes are swapped in order
	(2) Randomly select two nodes to swap order
	(3) Coding crossover with global optimum layer 1
Layer 2 encoding	(4) Randomly increase or decrease some random integer
	(5) Randomly select numbers within the full range of integers
	(6) Coding crossover with global optimum layer 2
Layer 3 encoding	(7) Randomized selection list of two nodes swapping order
	(8) Randomly select nodes to increase or decrease by some random integer
	(9) Coding crossover with global optimum layer 3

5 Numerical experiments

5.1 Instance description and results

We take the data from a port on 4 July 2023 as an example for our analysis. The quay shoreline extends for a total length of 800 m and is equipped with eight quay cranes, each with a working distance of 300 m. Fifteen ships arrive at the port successively, and their details, including the captain's information, arrival time, expected departure time, and the amount of cargo to be loaded and unloaded, are shown in Table 4. We wrote the code in Python 3.9 and executed it on a computer featuring a Core i9 2.50 GHz CPU, 16.0 GB of RAM, and a 64-bit Windows 11 operating system. The parameter settings are shown in Table 5.

Based on the above relevant parameters, we calculate the results as follows: the cost of quay crane service is 235,320 CNY, the cost of quay crane movement is 70,670 CNY, the cost of ship delay is 35,000 CNY, and the final total cost is 340,990 CNY. The final allocation plan of the ship is shown in Figure 5.

The quay crane scheduling plan is shown in Figure 6. We illustrate with the example of Quay Crane 1. Quay Crane 1 initially remains idle and then commences service for Ship 2 mid-way through its stay; subsequently, it transitions to serve Ship 4. After completing the service, the quay crane stops working. It resumes service at 00:00 the following day to assist Ship 11 and ceases operation upon completion of this service. Quay Cranes 5, 6, and 7 also implement time-variant allocation. To ensure that Ship 12 departs on schedule, they cease service after a designated period of unloading and loading for Ship 12, thereby reducing the operational costs associated with the quay cranes.

5.2 Q-learning algorithm for result optimization discussion

Section 4 describes our use of the Q-learning algorithm to select genetic operators, which enhances the search speed of the

TABLE 4 Information on arriving ships.

Ship name	Arrival time	Estimated time of departure	Length of ship (m)	Number of containers to be loaded and unloaded	Min number of quay cranes	Max number of quay cranes
V1	2023–07–04 04:00:00	2023–07–04 08:00:00	90	220	1	2
V2	2023–07–04 05:00:00	2023–07–04 11:00:00	150	440	1	3
V3	2023–07–04 07:30:00	2023–07–04 11:00:00	100	280	1	3
V4	2023–07–04 10:30:00	2023–07–04 15:30:00	120	400	1	3
V5	2023–07–04 12:00:00	2023–07–04 16:00:00	180	500	1	3
V6	2023–07–04 14:00:00	2023–07–04 18:30:00	150	420	1	3
V7	2023–07–04 18:30:00	2023–07–04 22:00:00	200	520	1	4
V8	2023–07–04 20:30:00	2023–07–04 23:00:00	89	180	1	3
V9	2023–07–04 22:30:00	2023–07–05 03:00:00	110	310	1	3
V10	2023–07–04 23:00:00	2023–07–05 02:00:00	98	200	1	3
V11	2023–07–05 00:00:00	2023–07–05 08:30:00	120	350	1	3
V12	2023–07–05 02:30:00	2023–07–05 09:30:00	190	470	1	4
V13	2023–07–05 09:30:00	2023–07–05 12:30:00	120	300	1	3
V14	2023–07–05 09:30:00	2023–07–05 17:00:00	180	430	1	3
V15	2023–07–05 09:30:00	2023–07–05 15:30:00	130	300	1	3

TABLE 5 Parameter settings.

Parameter	Value
Population	50
Maximum number of iterations	1,000
Work efficiency of quay cranes	30 TEU/h
Maximum greed rate	0.6
Q-learning algorithm discount rate	0.3
Q-learning algorithm learning rate	0.1
Cost of quay cranes movements	1910 CNY/times
Cost of quay cranes service from 8:00 to 17:00	1330 CNY/hour
Cost of quay cranes service at other times	1110 CNY/hour
Ship delay cost	7000 CNY/hour

algorithm. However, it is also possible to optimize the algorithm without using the Q-learning algorithm. To demonstrate the effectiveness of the Q-learning algorithm, we have designed two genetic operator selection methods for comparative analysis: random selection and selection using the Q-learning algorithm. We conducted five sets of experiments, with the number of quay cranes set at eight and a maximum of 3,000 iterations for the algorithm. Each operator's results were run five times. The experimental results are shown in Table 6.

The comparison results indicate that the performance gap between the two algorithms is minimal when the number of ships is low. However, as the number of ships increases, the optimization capability of the Q-learning-assisted algorithm is significantly superior to that of the random selection method. This is attributed to the fact that with a smaller number of ships, the algorithm's complexity is manageable. In contrast, as the number of ships grows, the random selection method

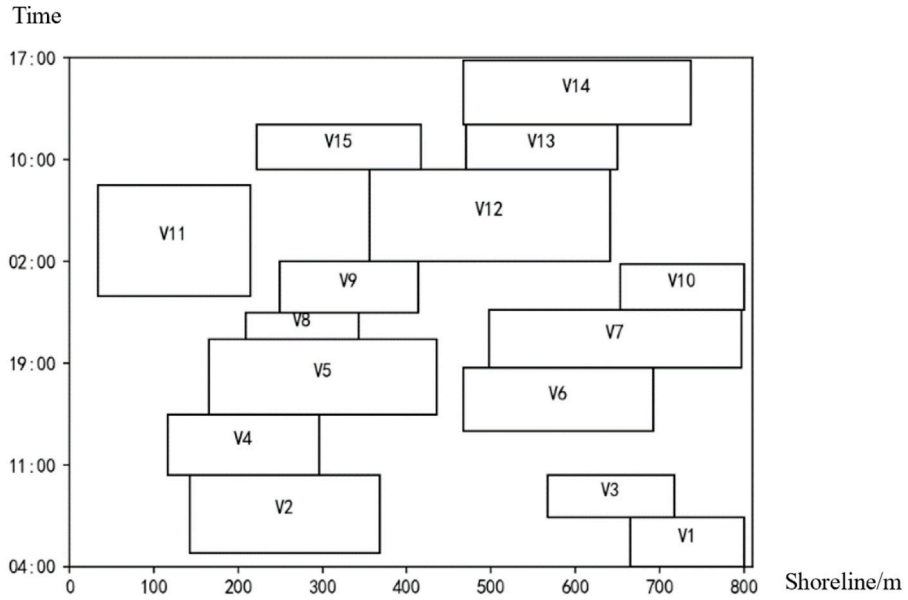

FIGURE 5
Ship allocation plan.

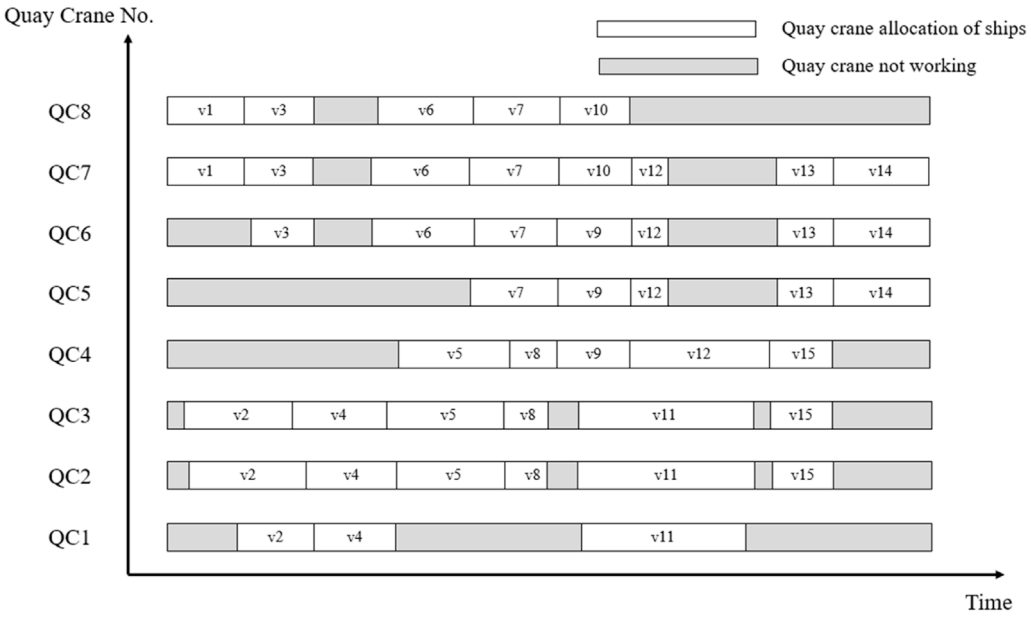

FIGURE 6
Quay crane allocation diagram.

introduces uncertainty and a higher likelihood of becoming trapped in local optima. The algorithm optimized with Q-learning also exhibits a reduced running time, demonstrating its superior search capabilities.

5.3 Algorithm effectiveness analysis

To verify the effectiveness of the algorithm presented in this article, we compare it with the CPLEX solver and genetic algorithm.

TABLE 6 Comparison of optimization with and without Q-learning.

Number of ships	No Q-learning optimization		With Q-learning optimization		
	Target value (CNY)	Running time (s)	Target value (CNY)	Running time (s)	GAP
10	23,311	112	23,392	61	−0.35%
20	41,675	168	40,648	82	2.53%
30	73,428	234	71,379	96	2.87%
40	85,301	312	82,301	116	3.65%
50	11,081	426	10,653	127	4.02%

TABLE 7 Performance of different algorithms in the examples.

Number of ships × Number of bridges	CPLEX		Genetic algorithm				Algorithms in this article			
	Cost (CNY)	Running times (s)	Average cost (CNY)	Optimal cost (CNY)	GAP	Running times (s)	Average cost (CNY)	Optimal cost (CNY)	GAP	Running times (s)
10 × 5	242,410	6,760	266,880	264,230	8.3%	87	248,180	245,720	1.3%	61
10 × 10	233,920	10,780	255,300	252,770	7.5%	83	238,690	236,320	1.0%	57
10 × 15	262,600	15,300	287,060	284,210	7.6%	88	268,410	265,750	1.2%	68
20 × 8	415,630	18,000	455,640	451,120	7.9%	103	421,910	417,730	0.5%	72
20 × 10	406,480	18,000	444,930	440,520	7.7%	112	413,820	409,720	0.8%	84
20 × 15	427,620	18,000	468,080	464,440	7.9%	114	433,810	429,510	0.4%	84
30 × 7	736,430	18,000	810,260	802,230	8.2%	138	739,800	732,470	−0.5%	106
30 × 10	713,790	18,000	784,160	776,390	8.1%	111	716,810	709,710	−0.6%	92
30 × 15	714,730	18,000	784,800	777,020	8.0%	119	715,800	708,710	−0.8%	117

The genetic algorithm adopts the traditional crossover and mutation logic and simultaneously performs crossover and mutation operations on the three layers of codes. We calculate the average of five runs for each algorithm under various scenarios, with the final results presented in Table 7. Given the difficulty of solving large-scale problems with CPLEX, we impose a 3-hour time limit on the solver, terminating it after this duration. The gap calculationformula is defined as follows, taking the genetic algorithm shown in Equation 34 as an example:

$$GAP = \frac{Obj_{\text{cplex}} - Obj_{GA}}{Obj_{GA}} \times 100\%. \tag{34}$$

From Table 7, it can be found that when the number of quay cranes is constant, as the number of ships increases, the overall cost and solution time of the ships will increase accordingly. This is because an increase in the number of ships leads to an increase in the complexity and workload of loading and unloading operations, which increases the overall cost and solution time. In terms of cost, our proposed algorithm demonstrates lower costs across all examples, with its optimal costs significantly lower than those of the genetic algorithm. In comparison with CPLEX, our algorithm has a significantly lower GAP value than the genetic algorithm, indicating that its cost is much closer to the optimal cost. In certain large-scale scenarios (for example, 30 × 5 and 30 × 10), our proposed algorithm's GAP is negative, indicating that its performance surpasses that of CPLEX, which is limited by time constraints. This further confirms the superior search capability of our algorithm in handling large-scale cases.

In Table 7, "Running time/s" refers to the computational time required to complete the optimization process, measured in seconds. This metric is used to evaluate the efficiency of the proposed method. The running time of our proposed algorithm is generally lower than that of the genetic algorithm and much lower than that of CPLEX, which reflects the high computational efficiency. Q-learning effectively reduces the running time by introducing a dynamic mechanism for operator selection. Q-learning leverages the performance of past iterations to dynamically adapt the selection of genetic operators. This ensures that only the most effective operators are utilized at each stage of the optimization process, thereby avoiding low-efficiency operations and reducing unnecessary computational effort. In summary, the algorithm proposed in this article offers significant advantages in terms of solution speed and accuracy.

6 Conclusion and future work

In this article, the joint berth and time-variant quay crane allocation problem is addressed. This is a cooperative allocation problem where the allocation of quay cranes can vary over time, allowing cranes to serve other ships even if a ship's operation is not yet complete. To solve this problem, a new MIP model is constructed, with the objective of minimizing ship delay costs, quay crane movement costs, and quay crane service costs while considering constraints such as quay crane working range and collision avoidance. A joint berth and time-variant quay crane allocation algorithm based on Q-learning is proposed, with a genetic algorithm as the main framework. The quay crane allocation module is embedded, and genetic operators are selected using the Q-learning algorithm. Q-learning is employed to evaluate the current population state and guide the generation of new solutions, enhancing optimization performance. This approach addresses the time-variant quay crane allocation problem by considering both crane and berth allocations to improve port resource utilization.

The results of the data analysis indicate that: 1) The method presented in this article can simultaneously address the continuous berth and time-variant quay crane allocation problems, reducing the total port cost. 2) The Q-learning selection module in this article can both accelerate the algorithm's convergence and enhance its search capability. 3) The algorithm proposed in this article outperforms the traditional genetic algorithm in terms of convergence speed and optimization ability in different scale examples. In small-scale examples, the proposed algorithm's performance is close to the exact solutions provided by CPLEX, and in some cases, it even surpasses the CPLEX algorithm when time constraints are applied, demonstrating the feasibility and superiority of our approach.

Future research could consider the following directions: 1) Incorporating the movement time of quay cranes into the model. 2) Extending the algorithm proposed in this article to include time-variant scheduling planning for automated guided vehicles, quay cranes, and berths within ports. 3) Future research may also incorporate the considerations of import and export container flows, along with a broader range of intricate factors that could influence port operations.

Data availability statement

The original contributions presented in the study are included in the article/supplementary material; further inquiries can be directed to the corresponding author.

Author contributions

CL: conceptualization, methodology, formal analysis, writing–original draft. DT: investigation, data curation, writing–original draft. RZ: investigation, methodology, writing–review and editing. YW: writing-review and editing, methodology, investigation.

Funding

The author(s) declare that financial support was received for the research, authorship, and/or publication of this article. National Natural Science Foundation of China (72271125) Shanghai Sailing Program (21YF1416400) Shanghai Rising-Star Program (21QB1404800).

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Generative AI statement

The author(s) declare that no Generative AI was used in the creation of this manuscript.

Publisher's note

References

Agra, A., and Oliveira, M. (2018). MIP approaches for the integrated berth allocation and quay crane assignment and scheduling problem. *Eur. J. Operational Res.* 264 (1), 138–148. doi:10.1016/j.ejor.2017.05.040

Chang, D., Jiang, Z., Yan, W., and He, J. (2010). Integrating berth allocation and quay crane assignments. *Transp. Res. Part E Logist. Transp. Rev.* 46 (6), 975–990. doi:10.1016/j.tre.2010.05.008

Correcher, J. F., and Alvarez-Valdes, R. (2017). A biased random-key genetic algorithm for the time-invariant berth allocation and quay crane assignment problem. *Expert Syst. Appl.* 89, 112–128. doi:10.1016/j.eswa.2017.07.028

Correcher, J. F., Alvarez-Valdes, R., and Tamarit, J. M. (2019). New exact methods for the time-invariant berth allocation and quay crane assignment problem. *Eur. J. Operational Res.* 275 (1), 80–92. doi:10.1016/j.ejor.2018.11.007

Hanagandi, V., and Nikolaou, M. (1998). A hybrid approach to global optimization using a clustering algorithm in a genetic search framework. *Comput. and Chem. Eng.* 22 (12), 1913–1925. doi:10.1016/s0098-1354(98)00251-8

Ji, B., Tang, M., Wu, Z., Yu, S. S., Zhou, S., and Fang, X. (2022). Hybrid rolling-horizon optimization for berth allocation and quay crane assignment with unscheduled vessels. *Adv. Eng. Inf.* 54, 101733. doi:10.1016/j.aei.2022.101733

Karam, A., and Eltawil, A. B. (2016). Functional integration approach for the berth allocation, quay crane assignment and specific quay crane assignment problems. *Comput. and Industrial Eng.* 102, 458–466. doi:10.1016/j.cie.2016.04.006

Krimi, I., Todosijević, R., Benmansour, R., Ratli, M., El Cadi, A. A., and Aloullal, A. (2020). Modelling and solving the multi-quays berth allocation and crane assignment problem with availability constraints. *J. Glob. Optim.* 78, 349–373. doi:10.1007/s10898-020-00884-1

Liang, C., Guo, J., and Yang, Y. (2011). Multi-objective hybrid genetic algorithm for quay crane dynamic assignment in berth allocation planning. *J. Intelligent Manuf.* 22, 471–479. doi:10.1007/s10845-009-0304-8

Liu, M. (2020). Research on port infrastructure, port efficiency and urban trade development. *J. Coast. Res.* 115 (SI), 220–222. doi:10.2112/jcr-si115-069.1

Lu, Z., Han, X., and Xi, L. (2011). Simultaneous berth and quay crane allocation problem in container terminal. *Adv. Sci. Lett.* 4 (6-7), 2113–2118. doi:10.1166/asl.2011.1533

Lujan, E., Vergara, E., Rodriguez-Melquiades, J., Jiménez-Carrión, M., Sabino-Escobar, C., and Gutierrez, F. (2021). A fuzzy optimization model for the berth allocation problem and quay crane allocation problem (BAP+ QCAP) with n quays. *J. Mar. Sci. Eng.* 9 (2), 152. doi:10.3390/jmse9020152

Malekahmadi, A., Alinaghian, M., Hejazi, S. R., and Assl Saidipour, M. A. (2020). Integrated continuous berth allocation and quay crane assignment and scheduling problem with time-dependent physical constraints in container terminals. *Comput. and Industrial Eng.* 147, 106672. doi:10.1016/j.cie.2020.106672

Meisel, F., and Bierwirth, C. (2009). Heuristics for the integration of crane productivity in the berth allocation problem. *Transp. Res. Part E Logist. Transp. Rev.* 45 (1), 196–209. doi:10.1016/j.tre.2008.03.001

Ng, W. C., and Mak, K. L. (2006). Quay crane scheduling in container terminals. *Eng. Optim.* 38 (6), 723–737. doi:10.1080/03052150600691038

Park, Y. M., and Kim, K. H. (2003). A scheduling method for berth and quay cranes. *OR Spectr.* 25 (1), 1–23. doi:10.1007/s00291-002-0109-z

Pham, V. H. S., Nguyen Dang, N. T., and Nguyen, V. N. (2024). Enhancing engineering optimization using hybrid sine cosine algorithm with Roulette wheel selection and opposition-based learning. *Sci. Rep.* 14 (1), 694. doi:10.1038/s41598-024-51343-w

Thanos, E., Toffolo, T., Santos, H. G., Vancroonenburg, W., and Vanden Berghe, G. (2021). The tactical berth allocation problem with time-variant specific quay crane assignments. *Comput. and Industrial Eng.* 155, 107168. doi:10.1016/j.cie.2021.107168

Türkoğulları, Y. B., Taşkın, Z. C., Aras, N., and Altınel, İ. K. (2014). Optimal berth allocation and time-invariant quay crane assignment in container terminals. *Eur. J. Operational Res.* 235 (1), 88–101. doi:10.1016/j.ejor.2013.10.015

Wang, Y. H., Li, T. H. S., and Lin, C. J. (2013). Backward Q-learning: the combination of Sarsa algorithm and Q-learning. *Eng. Appl. Artif. Intell.* 26 (9), 2184–2193. doi:10.1016/j.engappai.2013.06.016

Zheng, F., Pang, Y., Liu, M., and Xu, Y. (2020). Dynamic programming algorithms for the general quay crane double-cycling problem with internal-reshuffles. *J. Comb. Optim.* 39, 708–724. doi:10.1007/s10878-019-00508-9