

Two-level Crane Scheduling with Hybrid Metaheuristics in Realistic Port Operations

Hizia Amani, Linda Bouyaya, Rachid Chaib

Sustainable and efficient container unloading is a critical operation in maritime logistics, as it directly affects port throughput and ship turnaround time. Since quay cranes are the primary equipment responsible for this task, their effective scheduling is essential to minimize delays and ensure operational stability. This paper presents a two-level optimization approach for quay crane scheduling that balances operational efficiency with ship stability. At the bay level, we compare a flexible crane assignment strategy with the port's current strategy, which ensures ship stability but results in a long total unloading time. Using the LINGO solver and a Genetic Algorithm (GA), we achieve up to a 15% improvement. To leverage the port's strategy while overcoming its inefficiency, we redefine the unloading unit at the tier level and structure the process into two sequential phases. A hybrid metaheuristic combining the Firefly Algorithm (FA) and Cuckoo Search (CS) is proposed and compared with the GA. The hybrid algorithm shows higher consistency and statistically comparable solution quality to the GA.

KEY WORDS

- ~ Maritime transport
- ~ Port sustainability
- ~ Quay crane scheduling
- ~ Ship balance
- ~ Optimization
- ~ Metaheuristics

University of Constantine 1 - Frères Mentouri, Transport Engineering Department, Constantine, Algeria
e-mail: hizia.amani@doc.umc.edu.dz
doi: 10.7225/toms.v15.n02.w10

Received: 3 Jul 2025 / Revised: 26 Apr 2026 / Accepted: 5 Jun 2026 / Published: 20 Jun 2026

This work is licensed under



1. INTRODUCTION

Container traffic is a vital component of the global maritime supply chain, underpinning the flow of goods across continents with remarkable efficiency (Danladi et al., 2024). Its development has reshaped traditional logistics paradigms and strengthened the integrated relationships among key stakeholders in the maritime transport network, including shipping lines, port authorities, and hinterland connections (Elmi et al., 2022; Riadi et al., 2023). In recent decades, strategic efforts have focused on reducing overall shipping costs and minimizing the time ships spend along their transport routes, encompassing both port operations and inter-port navigation (Atacan et al., 2022; Kovalenko, 2024). Enhancing the performance of loading, unloading, and storage activities remains central to this goal (Ngoc et al., 2022; da Silva and Ensslin, 2024). To meet these objectives, the sector increasingly relies on advanced cargo tracking systems, automation of port functions, seamless communication among stakeholders, and optimized allocation of port resources (Zhang et al., 2025; Gao et al., 2023).

In recent years, sustainability has emerged as a strategic imperative in maritime logistics, shaping port design, operations, and policy (Jugovic et al., 2021; Zheng et al., 2023; Wilson et al., 2024). This shift reflects the broader ambition to reconcile efficiency with environmental responsibility (Giuffrida et al., 2021). Within this evolving context, safety and security are recognized as fundamental pillars of effective port management, especially during critical operations such as container unloading (Ding and Tseng, 2013).

Once a ship docks, quay cranes are deployed to unload containers in a sequence that requires careful coordination. Crane scheduling has therefore become a widely studied optimization problem, giving rise to a multitude of mathematical models focused primarily on minimizing the makespan, the total time required to complete all unloading tasks. However, despite the diversity and sophistication of these models, the issue of ship stability remains largely underexplored, as highlighted by Zhang et al. (2018). This gap persists even in studies addressing integrated loading and unloading (Eldemir & Taner, 2025), where balance constraints are not explicitly incorporated.

Ports, regardless of their size, infrastructure layout, handling capacity, or available equipment, require constant operational adjustment, particularly due to the strong interdependence among them. Ships during their maritime journeys often berth at multiple ports, each with its own logistical configuration. In this context, the current study builds upon established research on the Quay Crane Scheduling Problem (QCSP). In the previous studies, the decision unit was typically defined at the bay, bay area, or container-group level, mainly to present loading and unloading tasks. In contrast, this paper introduces a novel division of the ship into aft and forward sections, which leads to a different mathematical formulation and allows the model to account for the longitudinal balance of the ship. At the same time, adopting the tier as the unloading unit provides higher modeling flexibility, particularly for import-oriented operations, and helps preserve the transverse stability of the ship by ensuring a balanced distribution of unloading between the seaside and landside at each step of the operation. Compared to previous studies that do not consider stability at all, this approach ensures solutions that are feasible and safe in practice, reflecting real-world port operations and avoiding configurations that could be operationally unrealistic, such as cut-bay approaches that ignore the additional reshuffles often required during simultaneous loading and unloading operations. The model and operational practices considered in this study are directly inspired by procedures observed at Annaba Port- Algeria, particularly based on explanations from a port expert regarding container unloading operations. This source of technical insight supports the realism of the model while maintaining its general applicability.

The main highlights of this work are summarized as follows:

- This work addresses crane scheduling on two levels: the commonly used bay level and the less-explored tier level, adding depth to the modeling of container unloading.
- At the tier level, the proposed model implicitly considers the longitudinal balance of the ship, a constraint that has received limited attention in the existing literature.
- The firefly algorithm and Cuckoo Search were adapted to match the nature of the problem, resulting in solution quality statistically comparable to that of the genetic algorithm, while providing greater stability and consistency.

In the next section, we present the QCSP and its principal constraints addressed in the literature and highlight some papers that discuss ship stability during handling operations. In Section 3, we describe the problem at the two mentioned levels with its mathematical formulation. In Section 4, we present the algorithms used and explain how they were adapted. Results and discussion are provided in Section 5. Section 6 includes the conclusion and perspectives for future work.

2. QUAY CRANE SCHEDULING PROBLEM

The Quay crane scheduling problem is a critical issue in ports, as it directly impacts the efficiency of loading and unloading operations, which in turn affects the ship's stay time at the port (berthing time). This problem has received

significant attention from researchers, leading to the progressive incorporation of various operational constraints to improve the accuracy of scheduling models. We summarize below the key concepts relevant to the QCSP based on operational understanding and literature insights.

2.1. Key concepts

2.1.1. Interference & Safety margin

Quay cranes operate on rails, allowing them to move easily left or right. However, when multiple cranes are deployed simultaneously, they cannot overlap each other's working zones, as they are physically restricted from crossing paths. This constraint introduces significant challenges in scheduling crane operations, as planners must ensure that no two cranes interfere with each other's work areas to maintain efficiency and safety (Kim and Park, 2004; Chung and Chan, 2013).

The safety margin is the minimum permitted distance between two operating cranes to prevent operational conflicts and ensure safe handling. In general, studies and industry practices often consider a buffer of at least one bay between adjacent cranes, meaning that two cranes cannot operate simultaneously in directly adjacent bays (Moccia et al., 2006).

2.1.2. Container handling strategies

The configuration of ship and bay is illustrated in Figure 1. Container unloading strategies can significantly impact the efficiency of port operations. One commonly used method is bay-by-bay unloading, in which all containers from a specific bay are unloaded completely before moving on to the next bay. This systematic approach reduces unnecessary reshuffling and improves the predictability of operations, leading to smoother workflows and reduced crane idle times (Amani et al., 2023; Rodrigues and Agra, 2024).

Another widely adopted strategy is container clustering, which refers to the classification of containers based on predefined logistical criteria such as destination, cargo type, or handling requirements. The aim is to functionally align containers with similar downstream transport or processing needs, thereby improving coordination with inland transport and reducing transfer delays (Chen et al., 2014).

By contrast, container grouping concerns the physical arrangement of containers within the yard according to operational attributes such as size, weight or consignee. While clustering is logistics-oriented and decision-based, grouping is spatial and operational, focusing on stack stability, security considerations, and minimizing unnecessary reshuffling to enhance yard efficiency (Li and Li, 2022).

A related but distinct concept is priority-based unloading, which governs the sequence in which containers are discharged. Containers are processed based on urgency or risk; for example, refrigerated or high-value cargo may be unloaded first. Thus, prioritization determines when containers are handled, whereas clustering determines how they are classified, and grouping determines where they are placed (Zhang et al., 2018).

The choice of unloading method depends on multiple factors, including ship size, terminal capacity, and operational priorities. In highly automated ports, real-time data analysis is leveraged to determine the most efficient unloading strategy for each incoming ship. By integrating smart scheduling algorithms, terminals can dynamically adapt their unloading processes to reduce turnaround times and enhance overall productivity.

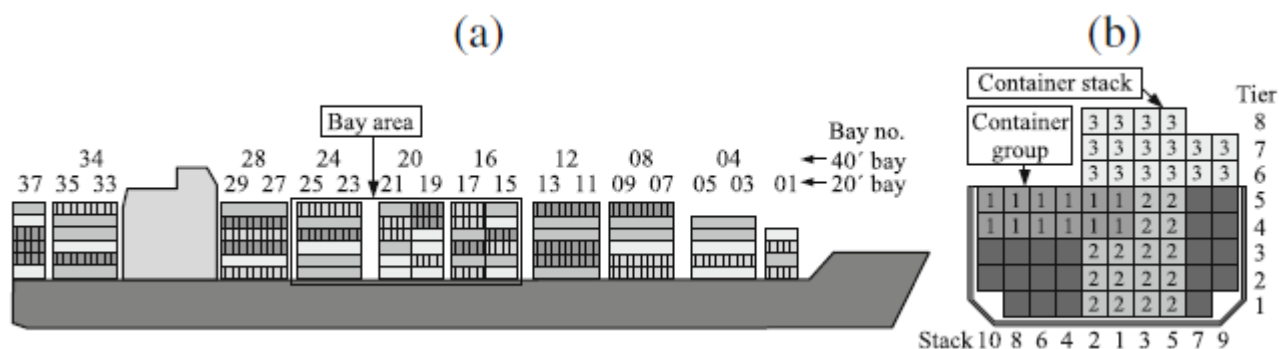


Figure 1. Ship (a) and Bay (b) configuration (Source: Bierwirth and Meisel, 2010).

2.1.3. Movement direction and Travel time

Unidirectional movement is often highlighted for its operational simplicity and efficiency. Under this mode, a crane consistently travels in a single direction along the ship, which facilitates straightforward scheduling and minimizes the time required to move from one bay to another. This predictable motion reduces the need for frequent stops and direction changes, thereby lowering transition times and improving overall throughput (Bierwirth, 2009; Chargui et al., 2023).

Bidirectional movement allows a quay crane to reverse its direction, offering enhanced flexibility in accessing containers located in varying sections of the ship. This flexibility may lead to a more effective allocation of crane tasks, especially in complex loading and unloading scenarios (Al-Dhaheri and Diabat, 2015).

Travel time refers to the duration a quay crane requires to move between its designated working positions along the ship. Shorter travel times allow for a higher number of container moves per unit time, contributing to both improved productivity and reduced operational costs (Wang and Kim, 2011).

2.1.4. Stability of the ship

Ship stability is a critical constraint in quay crane scheduling, as it directly affects ship safety at sea, within the port, and during loading and unloading operations. An imbalance in the distribution of weight can lead to severe consequences, including significant losses, ship capsizing, and potential loss of containers. Despite its importance, this constraint has traditionally received less attention in the literature compared to efforts aimed at minimizing makespan. Many researchers have primarily focused on optimizing crane scheduling to reduce overall operational time, while the stability issue has been more narrowly addressed. It is also worth noting that ship stability is a key consideration in the stowage problem (Van et al., 2024), where proper container arrangement is essential to maintaining the ship's equilibrium.

The stability of the ship is based on three concepts:

- Longitudinal stability: stability along the ship's length (bow to stern), preventing excessive trim (tilting forward or backward).
- Transversal stability: stability across the ship's width (port to starboard), preventing listing (leaning to one side).
- Vertical stability: stability related to the center of gravity and metacentric height (GM), ensuring the ship does not become top heavy and capsize.

2.2. Related works

Wang et al. (2013) were the first to address ship stability constraints. Their model ensures feasible loading and unloading sequences by restricting shifts in the ship's center of gravity. A steady-state genetic algorithm is developed to solve the problem, eliminating infeasible solutions while optimizing makespan. Ursavas (2017) proposed a model that integrates ship stability (longitudinal and transverse) during unloading operations, focusing on partial bay discharge at the port of Izmir, Turkey. However, partial discharging usually requires container relocation, which was not addressed in the study. Msakni et al. (2018) considered ship stability requirements and proposed a solution method that decomposes the original problem into two subproblems using column generation. Zhang et al. (2018) focused on ship longitudinal stability. They developed a MILP model and proposed a bicriteria evolutionary algorithm to minimize both makespan and maximum absolute trim. While the authors implicitly considered vertical ship stability by sequencing unloading from top to bottom and balancing laterally from the center outward, this general representation may not sufficiently reflect actual conditions during simultaneous loading and unloading operations, where the final container locations should be explicitly represented. Compared to previous studies, our work focuses on full unloading operations, which differ from models addressing partial discharge or mixed import–export scenarios. In addition, we adopt the tier level as the operational unit. These distinctions place this study in a different operational setting and provide a more detailed representation of the unloading sequence.

Hybrid metaheuristic methods have been applied to QCSP, though they remain relatively limited. Liang et al. (2011) proposed a multiobjective hybrid genetic algorithm for dynamic crane assignment. Guo et al. (2012) developed a hybrid generalized extremal optimization algorithm under interference constraints. Izquierdo et al. (2013) introduced a hybrid estimation-of-distribution algorithm (EDA) with local search. Cruz Reyes et al. (2016) integrated evolutionary search with neighborhood-based moves in a hybrid metaheuristic. Pérez Rodríguez (2021) returned to EDA, hybridizing it with a moth-flame heuristic. Amani et al. (2025) presented a hybrid harmony search with rule-based constraints to improve schedule quality and feasibility. In this work, Firefly and Cuckoo Search are chosen for their complementary strengths: Cuckoo Search provides broad exploration, while Firefly enhances local exploitation. Given that the tier-level formulation significantly enlarges the solution space, their combination is particularly well suited to tackling this complex problem.

3. PROBLEM DESCRIPTION

To capture the complexity of real unloading operations, this study adapts a two-level modeling approach. The first level, based on bay assignments, offers a simplified view commonly used in the literature. The second level introduces a more detailed tier-based formulation, allowing for greater flexibility, finer control over crane coordination, and better alignment with practical port constraints such as ship stability.

3.1. Bay-Level

This case considers a single container-ship made up of several bays. Each bay contains a number of containers to be unloaded. Two quay cranes are used for the operation. One crane moves on a fixed rail along the ship, while the other is mobile and can move more freely. Each bay is handled by only one crane, and a crane can move to another bay only after finishing the current one. The travel time between bays is not considered, this assumption is reasonable for terminals with small bay spacing where the crane's operational handling time dominates its movement time. Nevertheless, for large ship, including travel time can offer a more realistic representation of operations. Figure 2 illustrates the nature of the problem, the assignment of bays to the two quay cranes. Each crane can be allocated any set of bays, not necessarily consecutive, as long as the overall makespan is minimized.

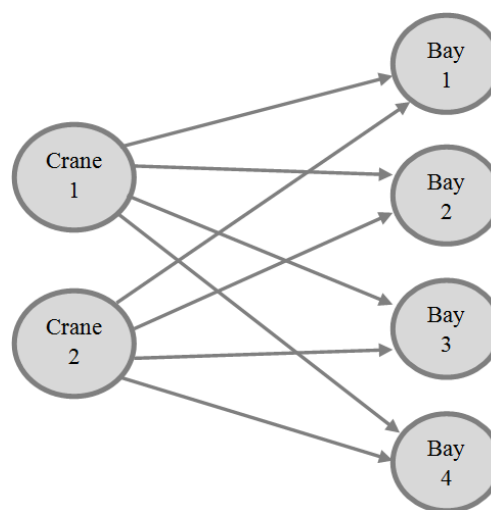


Figure 2. Assignment possibilities of bays to two quay cranes.

To formalize the bay-level problem, a mathematical model is developed to minimize the makespan. The following defines the input data, decision variables, and constraints involved.

3.1.1. Mathematical Formulation

Set

N: set of bays indexed i

Input data

N_i : number of containers in each bay.

U_1 : unloading time of one container by crane 1.

U_2 : unloading time of one container by crane 2.

Decision Variables

$X_i \in \{0, 1\}$: 1 if bay i is unloaded by crane 1; 0 otherwise.

$Y_i \in \{0, 1\}$: 1 if bay i is unloaded by crane 2; 0 otherwise.

C_1 : completion time of crane 1

C_2 : completion time of crane 2

Objective

$$\min \max (C_1, C_2) \dots \dots \dots (1)$$

S.t

$$\sum_{i \in N} X_i N_i U_1 = C_1 \dots \dots \dots (2)$$

$$\sum_{i \in N} Y_i N_i U_2 = C_2 \dots\dots\dots (3)$$

$$X_i + Y_i = 1 \quad \forall i = 1, \dots, n \dots\dots\dots (4)$$

(1): The objective function minimizes the makespan, defined as the maximum between the two cranes' completion times. (2) and (3): These constraints calculate the completion time of crane 1 and 2. (4): Each bay must be unloaded by only one crane.

3.2. Tier Level

Based on the port's unloading strategy, two cranes start from opposite ends of the ship and move toward the middle, to maintain longitudinal balance. However, this often creates significant discrepancies in crane completion times. To overcome this issue, we propose a two-phase unloading strategy

In the first phase, each crane unloads selected tiers in its respective section. At the forward section (F), bay 1 is located at the extreme left, at the aft section (A); bay 1 is located at the extreme right. Within each bay (j and k), tier 1 represents the top tier.

In the second phase, the cranes switch sections and complete the unloading of the remaining tiers. Figure 3 provides a visual representation of the ship's structure, illustrating both longitudinal and transverse sections relevant to the tier-level unloading model.

The formulation of this extended model is based on a practical approach observed in port operations, with necessary adjustments introduced to balance crane workloads across both phases.

To translate this practical unloading strategy into a formal model, the input data, decision variables, and constraints are presented in the following formulation.

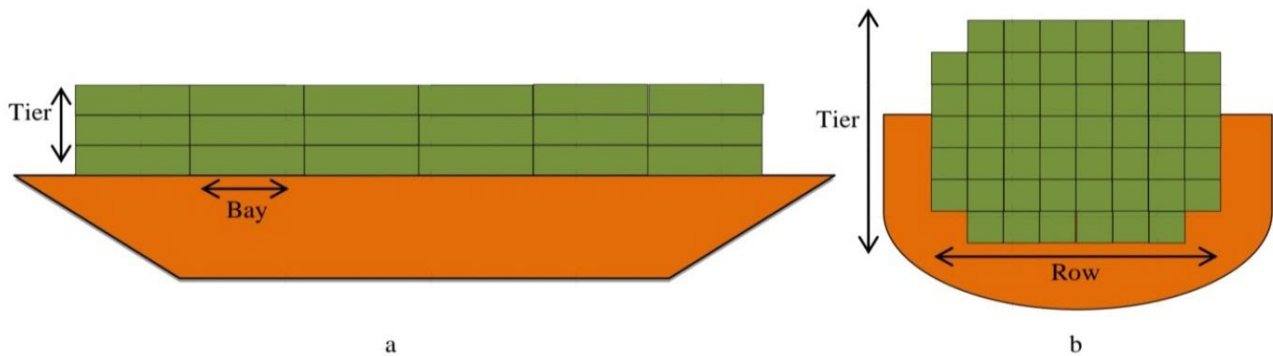


Figure 3. Longitudinal and transverse sections of the ship

3.2.1. Mathematical Formulation

Set

F: set of bays from bow to midship indexed by j

A: set of bays from stern to midship indexed by k

Subset

$M \subseteq F$: subset of tiers in forward bays j indexed by e

$N \subseteq A$: subset of tiers in aft bays k indexed by f

Input data

T_j : number of tiers on bay j

T_k : number of tiers on bay k

N_{ce} : number of containers in tier e

N_{cf} : number of containers in tier f

S_x : speed of crane 1

S_y : speed of crane 2

Decisions variables

$I_{xej} \in \{0, 1\}$: 1 if crane 1 unloads tier e on bay j; 0 otherwise

$Z_{yej} \in \{0, 1\}$: 1 if crane 2 unloads tier e on bay j; 0 otherwise

$V_{xfk} \in \{0, 1\}$: 1 if crane 1 unloads tier f on bay k; 0 otherwise

$U_{yfk} \in \{0, 1\}$: 1 if crane 2 unloads tier f on bay k; 0 otherwise

CP1: Cmax in phase 1
 CP11: Completion time of crane 1 in phase 1
 CP12: Completion time of crane 2 in phase 1
 CP2: Cmax in phase 2
 CP21: Completion time of crane 1 in phase 2
 CP22: Completion time of crane 2 in phase 2
Objective

$$\min \max (CP1 + CP2) \dots \dots \dots (5)$$

S.t

$$I_{xej} + Z_{yej} \leq 1 \quad \forall e \in M \quad \forall j \in F \dots \dots \dots (6)$$

$$V_{xfk} + U_{yfk} \leq 1 \quad \forall f \in N \quad \forall k \in A \dots \dots \dots (7)$$

$$\sum_{e \in M} (I_{xej} + Z_{yej}) = Tj \quad \forall j \in F \dots \dots \dots (8)$$

$$\sum_{f \in N} (V_{xfk} + U_{yfk}) = Tk \quad \forall k \in A \dots \dots \dots (9)$$

$$I_{xej} \geq I_{xej'} \quad \forall e \in M \dots \dots \dots (10)$$

$$Z_{yej} \geq Z_{yej'} \quad \forall e \in M \dots \dots \dots (11)$$

$$V_{xfk} \geq V_{xfk'} \quad \forall f \in N \dots \dots \dots (12)$$

$$U_{yfk} \geq U_{yfk'} \quad \forall f \in N \dots \dots \dots (13)$$

$$\sum_{e \in M} I_{xej} H_{ej} = CP11 \dots \dots \dots (14)$$

$$\sum_{f \in N} U_{yfk} H_{fk} = CP12 \dots \dots \dots (15)$$

$$\sum_{e \in M} Z_{yej} H_{fk} = CP21 \dots \dots \dots (16)$$

$$\sum_{f \in N} V_{xfk} H_{fk} = CP22 \dots \dots \dots (17)$$

$$\max (CP11, CP12) = CP1 \dots \dots \dots (18)$$

$$\max (CP21, CP22) = CP2 \dots \dots \dots (19)$$

$$H_{ej} I_{xej} \leq N_{ej} S_x \quad \forall e \in M \quad \forall j \in F \dots \dots \dots (20)$$

$$H_{ej} Z_{yej} \leq N_{ej} S_y \quad \forall e \in M \quad \forall j \in F \dots \dots \dots (21)$$

$$H_{fk} V_{xfk} \leq N_{fk} S_x \quad \forall f \in N \quad \forall k \in A \dots \dots \dots (22)$$

$$H_{fk} U_{yfk} \leq N_{fk} S_y \quad \forall f \in N \quad \forall k \in A \dots \dots \dots (23)$$

$$I_{xej}, V_{xfk}, Z_{yej}, U_{yfk} \in \{0, 1\} \dots \dots \dots (24)$$

$$CP1, CP2, CP11, CP12, CP21, CP22 \geq 0 \dots \dots \dots (25)$$

(5): The objective function minimizes the makespan in both phases. (6) and (7): Each tier must be assigned to at most one crane. (8) and (9): All tiers in each bay must be unloaded. (10) to (13): These constraints ensure that cranes unload tiers sequentially, starting from the top and progressing downward without skipping any intermediate tier. Here, e' and f' denote the immediate successive tiers of e and f , respectively. (14) to (17): These constraints calculate the individual completion time of each crane during both unloading phases. (18) and (19): These constraints determine the makespan (Cmax) of phase 1 and phase 2, by selecting the maximum completion time between the two cranes. (20) to (23): These constraints compute the handling time for each tier assigned to the cranes during both phases. (24) and (25): These constraints define the binary decision variables and ensure non negativity for the completion time variables.

In the next section, we define and explain the algorithms used to solve the problem at both bay and tier levels.

4. RESOLUTION METHODS

To solve the objective function (1), we use the LINGO solver and a genetic algorithm developed in Python 3.9.

LINGO is a software tool designed to efficiently build and solve linear, nonlinear, and integer optimization models. We have created a simple LINGO model to solve the problem.

Python is an interpreted object- oriented, high-level programming language with dynamic semantics.

4.1. Genetic Algorithm

GA is a search and optimization technique inspired by the process of natural evolution, where a population of solutions evolves over generations through selection, crossover, and mutation to find near-optimal solutions to complex problems (Holland, 1992; Forrest, 1996; Miller and Goldberg, 1995).

4.1.1. Chromosome representation

The chromosome representation used in the genetic algorithm consists of a sequence whose length equals the number of bays in the ship. Each gene takes a value of either 1 or 2, representing assignment to crane 1 or crane 2, (Figure 4). The bays are numbered from left to right (from bow to stern), and this order remains fixed across all chromosomes. We select random individuals from the population, apply one-point crossover, and then perform swap mutation.

Bay number	1	2	3	4	5	6	7	8
Crane number	1	1	1	2	1	2	2	2

Figure 4. Chromosome representation

To solve the objective function (5) effectively, we used a hybrid algorithm combining the Firefly Algorithm (FA) and Cuckoo Search (CS), implemented in Python 3.9.

4.2. Firefly algorithm

The firefly algorithm is a nature-inspired metaheuristics based on the flashing behavior of fireflies. It assumes that each firefly is attracted to others based on their brightness, which is related to the fitness of the solution they represent (Yang, 2009; Dong et al. 2018). The basic steps of FA include:

Attraction Rule: a less bright firefly moves towards a brighter one based on the distance and attractiveness. Attractiveness Function: given by:

$$\beta(r) = \beta_0 e^{(-\gamma r^2)} \dots\dots\dots (26)$$

Where:

β_0 : is the maximum attractiveness.

γ : is the absorption coefficient.

r : is the distance between two fireflies.

Movement Equation: the movement of firefly (i) towards a brighter firefly (j) is given by:

$$x_i = x_i + \beta(r)(x_j - x_i) + \alpha \dots\dots\dots (27)$$

Where:

α : is a randomization parameter.

The random step introduces stochasticity for exploration.

Discrete adaptation of the firefly algorithm

In our discrete problem, the smallest possible move is always a change of one tier. Therefore, instead of the continuous attractiveness decay, we fix a minimum attractiveness value ($\beta_{\min} = 0.3$) to maintain gradual improvement. The distance (r) is defined as the number of bays whose tier-boundaries differ between two solutions. the number of bays updated in each move is given by:

$$K = \text{round}(\beta_{\min} \times r)$$

If $r = 1$: we directly modify the differing bay by ± 1 tier.

If $r > 1$: we update K bays selected from the differing set.

For $K = 1$: each candidate bay is tested separately, and the best result is kept.

For $K > 1$: combinations of K bays are evaluated, and the update that yields the best solution is selected.

Example

If the guiding solution has a bay boundary of 3 and the current solution has 1, we increase the tier to 2. Conversely, if the current boundary is 5, we decrease it to 4.

4.3. Cuckoo Search Algorithm

The Cuckoo Search is another nature-inspired algorithm based on the brood parasitism of certain cuckoo species. It uses Levy Flight to generate new solutions far from the current ones, providing a powerful global search capability (Gandomi et al., 2013; Yachba et al. 2020). Key components include:

Random walk with Levy Flight:

$$x_{t+1} = x_t + Levy(\alpha, \beta)$$

Where:

The step size is governed by the levy distribution, allowing occasional long jumps for exploration.

Big jump with Levy Flight: this is used to replace the fixed probability jump mechanism, allowing the algorithm to adapt based on the current solution's distance from the global best.

Levy Flight: is a random walk where the step lengths have a heavy-tailed probability distribution, making long jumps more frequent. The mathematical model for Levy Flight is given by:

$$L(\alpha, \beta) \sim \frac{\Gamma(1+\alpha) \sin \frac{\pi\alpha}{2}}{\Gamma(\frac{1+\alpha}{2}) \alpha 2^{(\alpha-1)/2}} \cdot \frac{1}{|v|^{1/\alpha}} \dots\dots\dots (28)$$

Where:

α : controls the step size.

β : scales the step intensity.

Γ : is the Gamma function.

Discrete adaptation of the Cuckoo Search

In our discrete setting, we distinguish between two types of moves:

Big jump: this corresponds to a radical modification of bay boundaries. For each selected bay, the boundary is completely reassigned to new random value within the feasible range.

Small jump: this corresponds to a light adjustment, in selected bays, we add or remove exactly one tier (± 1), not guided toward the best individual (unlike firefly Algorithm).

We denote J as the length of the jump which is the number of selected bays to be changed, this length is applied for small and big jump, and it is:

$1 \leq J \leq B$; where B is the total number of bays. Thus, the minimum jump is a modification in a single bay, and the maximum jump may involve all bays simultaneously.

How J is drawn (Pareto heavy-tailed law)

We sample a uniform random number $U \sim \text{Uniform}(0, 1)$ and map it via a Pareto distribution so that long jumps are more frequent:

$$s = (1 - U)^{(-1/\lambda)}$$

$$J = \min \{B, \max (1, \text{round}(s))\}$$

We set $\lambda = 1$, with this setting, the longest jump generated by levy flight reached approximately 10, which corresponds to about ten bays. This value can be considered adequate for our problem context.

4.4. Solution representation

To address the problem of crane assignment to tiers, a solution structure is developed to facilitate the exploration of the solution space. In this context, the boundary of a crane is defined as a key component that specifies, for each crane, the exact number of tiers it can unload from bays j and k during phases 1 and 2.

As illustrated in Figure 5, the ship consists of 8 bays: 4 in the aft section and 4 in the forward section. The number of tiers in each bay is 4, 6, 5, 5, 5, 4, 6, and 4, respectively. The first boundary value of crane 1 is 2, meaning that this crane unloads up to tier 2 (i.e., tier 1 and tier 2) during phase 1. For example, bay 1 has 4 tiers; since crane 1 unloads up to tier 2, the remaining tiers (tier 3 and tier 4) are assigned to crane 2 in phase 2. The boundary values in section F correspond to crane 1, while the boundary values in section A correspond to crane 2.

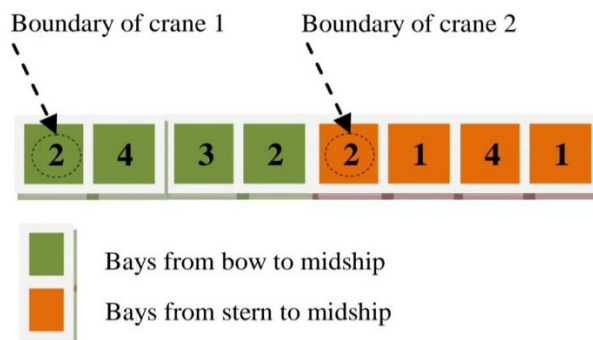


Figure 5. Solution representation

4.5. The pseudo-code of the Hybrid Algorithm

Input:

N	→ Population size
MaxIter	→ Maximum number of iterations
Beta_min	→ Maximum attractiveness parameter
Pareto_lambda	→ Lévy/Pareto distribution parameter
P_big_jump	→ Probability of large jump
Problem_data	→ Discrete combinatorial instance

Output:

S_best	→ Best feasible solution
F_best	→ Best objective value

-
- 1: Generate N feasible solution
 - 2: Evaluate fitness $f(S_i)$ for each solution S_i
 - 3: $S_best \leftarrow \operatorname{argmin} f(S_i)$
 - 4: For iter = 1 to MaxIter do
 - 5: Sort population in ascending order of fitness
 - 6: Elite $\leftarrow$ Best solution in population
 - 7: **// Intensification Phase – Discrete Firefly (Top 30%) //**
 - 8: For each solution S_i in Top 30% (excluding Elite) do
 - 9: Identify discrete differences between S_i and S_best
 - 10: Compute $k \leftarrow \max(1, \text{round}(\text{beta_min} \times \text{number_of_differences}))$
 - 11: Select k positions and move one discrete step towards S_best
 - 12: Repair solution if constraints violated
 - 13: If $f(\text{new_}S_i) < f(S_i)$ then
 - 14: $S_i \leftarrow \text{new_}S_i$
 - 15: End If
 - 16: End For
 - 17: **// Diversification Phase – Discrete Cuckoo (Bottom 70%) //**
 - 18: For each solution S_j in bottom 70% do
-

19:	Determine jump length J using Pareto distribution
20:	Randomly select J discrete positions
21:	For each selected position do
22:	If rand () < P_big_jump then
23:	Assign random feasible discrete value
24:	Else
25:	Apply ± 1 bounded discrete move
26:	End If
27:	End For
28:	Repair solution if constraints violated
29:	If f (new_Sj) < f(Sj) then
30:	Sj $\leftarrow$ new_Sj
31:	End If
32:	End For
33:	Reinsert Elite if lost
34:	Update S_best $\leftarrow$ argmin f (Si)
35:	End For
36:	Return S_best, f_best

The hybrid algorithm starts by generating an initial population of feasible solutions and evaluating their fitness values to identify the current best solution. At each iteration, the population is sorted in ascending order of fitness. The top 30% of solutions undergo an intensification phase inspired by the discrete Firefly Algorithm, where each solution is partially moved toward the global best by modifying a subset of differing positions, followed by feasibility repair and acceptance if improvement is achieved. The remaining 70% of solutions enter a diversification phase based on the discrete Cuckoo Search, where Lévy/Pareto-distributed jump lengths determine the number of positions to modify through either large random reassignment or bounded $\pm$ adjustments. Elitism is enforced to preserve the best solution, and the process repeats until the maximum number of iterations is reached. The algorithm finally returns the best feasible solution and its corresponding objective value.

5. RESULTS

To evaluate the performance of the proposed model at the bay level, a comparative analysis was carried out using three different solutions: the LINGO solver, the genetic algorithm, and the port strategy. Although the port strategy was initially described in the context of tier-level unloading, its principles were applied here due to their operational similarity to the bay-by-bay unloading commonly observed in practice. The goal of this comparison is to assess the efficiency of each method in minimizing the overall unloading time (makespan). The results for one ship with 3 to 9 bays are summarized in Table 1.

To assess the stability of the proposed hybrid algorithm (HY), multiple independent executions were performed on two distinct problem instances. The first instance consists of 4 bays (NB) and 6 tiers (NT) per bay, while the second includes 6 bays and 6 tiers. Tables 2 and 3 present the makespan values (per hour) obtained across five separate runs for each case, comparing the results with those produced by the genetic algorithm. This evaluation aims to verify the consistency of solutions under the inherent randomness of metaheuristic algorithms. To further evaluate the performance and scalability of the proposed hybrid algorithm, a series of instances were tested involving ships with 8, 10, and 12 bays. Table 4 summarizes the makespan and CPU time obtained for each instance, comparing the hybrid approach with the genetic algorithm. The objective is to assess not only solution quality but also computational efficiency as the problem size increases. Due to the stochastic nature of evolutionary algorithms, the GA was executed in multiple independent runs for each instance, and the best makespan obtained was recorded.

For the hybrid Algorithm and the genetic algorithm, a population size of 30 and 40 iterations were used for instances with 8 to 10 bays, while a population size of 40 and 50 iterations was used for instances with 12 bays. For the GA, the tournament selection method with a size of 3 was applied, with a mutation rate of 0.05 and a crossover rate of 0.8. Elitism was also employed.

Instances	Number of bays	GA (min)	Lingo (min)	Port strategy (min)	Gap %
1	3	148	148	156	5.13
2	4	222	222	234	5.13
3	5	236	236	237	0.42
4	6	282	282	320	11.9
5	7	321	321	378	15.1
6	8	376	376	390	3.59
7	9	423	423	454	6.83

Table 1. Comparison of Bay-Level Results Using GA and LINGO Solver

Execution	NB	NT	HY (h)	GA (h)
1	4	6	5.2	5.28
2	4	6	5.13	5.4
3	4	6	5.13	5.13
4	4	6	5.13	5.2
5	4	6	5.13	5.2

Table 2. Stability Analysis of Tier-Level (Instance: 4 bays, 6 tiers)

Execution	NB	NT	HY	GA
1	6	6	7.53	7.7
2	6	6	7.62	7.76
3	6	6	7.6	7.85
4	6	6	7.62	7.86
5	6	6	7.6	7.85

Table 3. Stability Analysis of Tier-Level (Instance: 6 bays, 6 tiers)

Execution	NB	NT	Makespan (H)		CPU time (S)	
			HY	GA	HY	GH
1	8	6	10.33	10.53	<1	<1
2	8	6	10.07	9.78	<1	<1
3	8	6	9.82	9.31	<1	<1
4	8	6	9.65	9.53	<1	<1
5	10	7	15.03	15.43	2.543	<1
6	10	7	15.33	15.8	2.605	<1
7	10	7	15.02	15.46	2.543	<1
8	10	7	14.85	15.5	2.589	<1
9	12	8	26.05	26.18	7.036	<1
10	12	8	26.00	26.13	7.238	<1
11	12	8	26.17	26.52	7.067	<1
12	12	8	26.15	26.47	7.13	<1

Table 4. Performance Comparison on Extended Tier-Level Instance

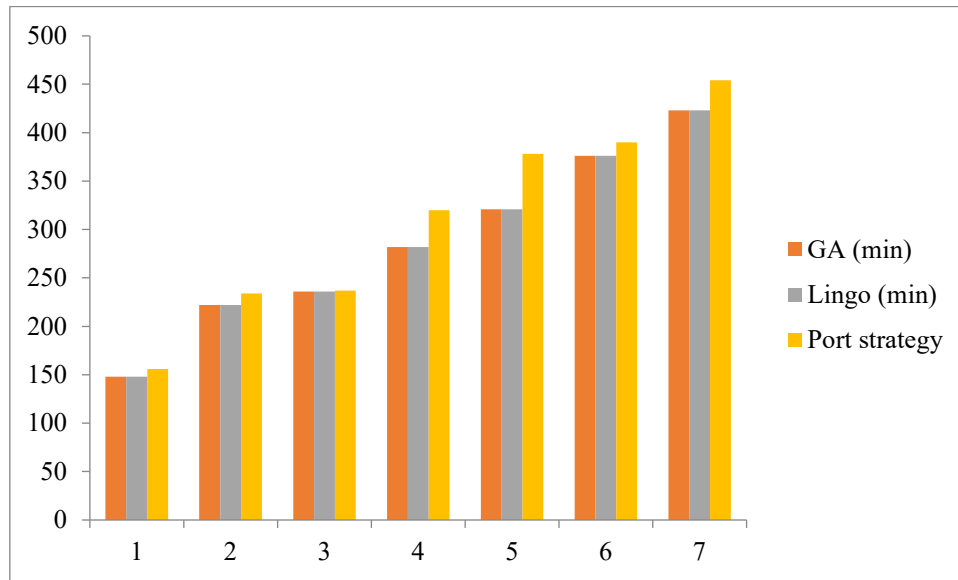


Figure 6. Comparison of makespan results using GA, LINGO, and port strategy for different bay numbers

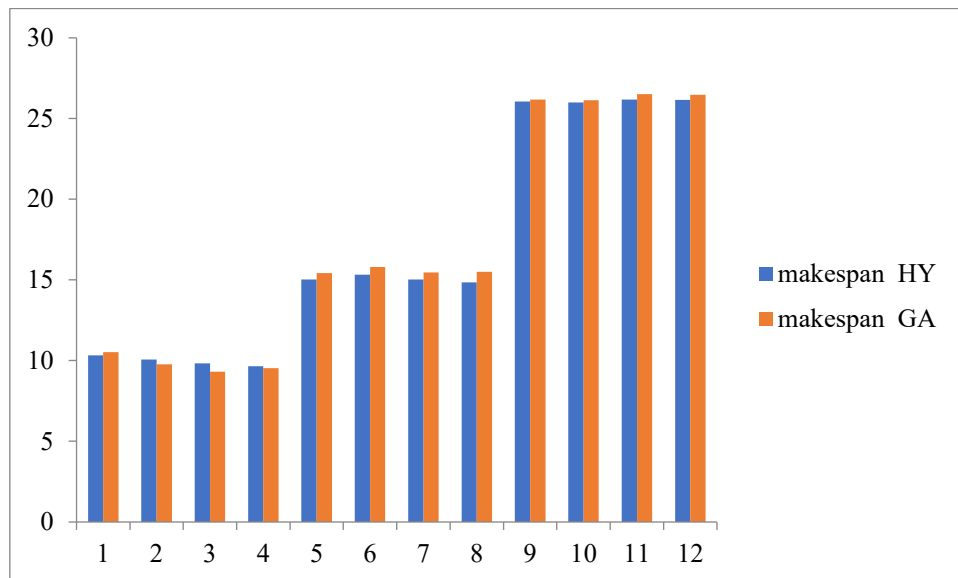


Figure 7. Comparison of makespan results between the hybrid FA+CS and GA for various problem instances

5.1. Discussions

As presented in Table 1 and Figure 6, the unloading time obtained by LINGO solver and the genetic algorithm are identical across all tested instances, which are expected given the small size of the problem.

The port strategy is efficient because cranes progress through adjacent bays, minimizing two-direction travel. However, it enforces a fixed partition of bays per crane. With uneven container distribution across bays, this lack of flexibility increases completion time. In our small-ship cases (9 bays), Lingo and GA improves over the port baseline by 0.4% - 15%, confirming that optimal sequencing outperforms the fixed split when loads are unbalanced.

As observed in Tables 2 and 3, both the genetic algorithm and the hybrid approach achieve close makespan values across multiple runs. For the instance with 4 bays and 6 tiers (Table 2), the hybrid algorithm maintained a stable makespan between 5.13 and 5.2 (≈ 308 - 312 minutes), while the GA ranged from 5.13 to 5.4 (≈ 308 - 324 minutes). Similarly, for the second instance (6 bays and 6 tiers, Table 3), the hybrid remained within 7.53 to 7.62 (≈ 452 - 457 minutes), while the GA fluctuated from 7.7 to 7.86 (≈ 462 - 472 minutes).

Although the differences appear small, they highlight a consistent pattern: the hybrid algorithm produces tightly grouped results with minimal deviation, while GA exhibits higher variability. For the 4_bay instance, the hybrid approach achieved a standard deviation of 0.031 h (≈ 1.86 min), compared to 0.102 h (≈ 6.12 min) for GA. Similarly, in the 6-bay

instance, the hybrid recorded 0.037 h (≈ 2.22 min) versus 0.075 (≈ 4.50 min) for GA. These results quantitatively confirm the superior stability of the hybrid method across repeated runs.

As illustrated in Figure 7 and detailed in Table 4, the makespan values obtained by the HY and the GA are closely aligned across all extended instances. To statistically validate this observation, a paired comparison was conducted. The normality of paired differences was verified using the Shapiro-Wilk test ($p = 0.432$), supporting the use of a paired t-test. The results indicated no statistically significant difference between the two approaches ($t(11) = -1.85$, $p = 0.091$). The mean makespan was 17.04 hours for HY and 17.22 hours for GA, with a mean difference of -0.18 hours (95% CI [-0.396, 0.034]). A Wilcoxon signed-rank test yielded consistent finding ($p = 0.092$). Although the difference did not reach the conventional 0.05 significance threshold, the effect size was moderate (Cohen's $d_z = -0.53$), suggesting a practically relevant tendency favoring the hybrid approach.

Although GA requires less than one second of CPU time, the hybrid algorithm reaches up to 7 seconds due to its additional intensification and diversification mechanisms. However, since scheduling is performed per ship and the makespan is measured in hours, this difference is practically negligible. Statistical results show comparable solution quality, while the hybrid algorithm demonstrates greater stability, justifying its slightly higher computational effort.

6. CONCLUSION

In this study, a two-level optimization framework for the Quay Crane Scheduling Problem was proposed to progressively bridge operational efficiency and ship stability considerations. At the bay level, flexible crane assignment was first investigated and compared with the port's fixed strategy. The results showed that optimization through exact and metaheuristic approaches can reduce the makespan by up to 15%, highlighting the limitations of rigid operational splits.

Building on these findings, the model was then extended to the tier level, where the unloading unit was redefined to provide finer operational control and better alignment with practical stability requirements. This second level preserves the operational realism of the port strategy while overcoming its inefficiency through a structured two-phase process. The proposed hybrid Firefly-Cuckoo Search Algorithm achieved statistically comparable solution quality to the genetic algorithm, while demonstrating greater consistency across runs.

Although stability was handled implicitly in our formulation, explicit constraints remain beneficial in certain operational contexts, particularly when loading and unloading occur simultaneously and the ship's center of gravity must be carefully monitored. Moreover, partial unloading operations often require container reallocation to preserve stability, which highlights the practical relevance of incorporating such constraints.

Future work could extend the approach to jointly consider ships and trucks, leveraging the flexibility of tier-level and two-phase unloading strategies. However, since ships may be served by a variable number of cranes, scaling the framework would require more adaptive and robust scheduling approaches.

CONFLICT OF INTEREST

Authors declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

REFERENCES

- Al-Dhaheer, N. and Diabat, A. (2015) 'The quay crane scheduling problem', *Journal of Manufacturing Systems*, 36, pp. 87-94.
- Amani, H., Bouyaya, L. and Chaib, R. (2025) 'A hybrid Harmony Search and Rule-Based Approach for Dynamic Crane Scheduling Problem', *Journal of ETA Maritime Science*.
- Amani, H., Bouyaya, L., Chaib, R., Djekrif, F. Z. and Aizi, M. (2022) 'Optimisation of Quay Crane Scheduling Problem at the Port of Algeria', in *International Conference on Reliability and Statistics in Transportation and Communication*. Cham: Springer International Publishing, pp. 232-241.
- Atacan, C., Kayıran, B. and Açıık, A. (2022) 'Impact of liner shipping connectivity on container traffic in Turkish ports', *Transactions on Maritime Science*, 11(2), pp. 2-2.
- Bierwirth, C. and Meisel, F. (2009) 'A fast heuristic for quay crane scheduling with interference constraints', *Journal of Scheduling*, 12, pp. 345-360.
- Bierwirth, C. and Meisel, F. (2010) 'A survey of berth allocation and quay crane scheduling problems in container terminals', *European Journal of Operational Research*, 202(3), pp. 615-627.

- Chargui, K., Zouadi, T., Sreedharan, V. R., El Fallahi, A. and Reghioui, M. (2023) 'A novel robust exact decomposition algorithm for berth and quay crane allocation and scheduling problem considering uncertainty and energy efficiency', *Omega*, 118, 102868.
- Chen, J. H., Lee, D. H. and Goh, M. (2014) 'An effective mathematical formulation for the unidirectional cluster-based quay crane scheduling problem', *European Journal of Operational Research*, 232(1), pp. 198-208.
- Chung, S. H. and Chan, F. T. (2013) 'A workload balancing genetic algorithm for the quay crane scheduling problem', *International Journal of Production Research*, 51(16), pp. 4820-4834.
- da Silva, J. C. and Ensslin, S. (2024) 'Performance evaluation in the port sector: A systematic literature review', in *Maritime Transport Conference* (No. 10).
- Danladi, C., Tuck, S., Tziogkidis, P., Tang, L. and Okorie, C. (2024) 'Efficiency analysis and benchmarking of container ports operating in lower-middle-income countries: a DEA approach', *Journal of Shipping and Trade*, 9(1), p. 7.
- Ding, J. F. and Tseng, W. J. (2013) 'Fuzzy risk assessment on safety operations for exclusive container terminals at Kaohsiung port in Taiwan', *Proceedings of the Institution of Mechanical Engineers, Part M: Journal of Engineering for the Maritime Environment*, 227(2), pp. 208-220.
- Dong, L., Yang, Y. and Sun, S. (2019) 'QCs scheduling scheme of genetic algorithm (GA) and improved firefly algorithm (FA)', *Cluster Computing*, 22, pp. 4331-4348.
- Eldemir, F. and Taner, M. E. (2025) 'A Hybrid Meta-Heuristic Approach for Solving Single-Vessel Quay Crane Scheduling with Double-Cycling', *Journal of Marine Science and Engineering*, 13(2), p. 371.
- Elmi, Z., Singh, P., Meriga, V. K., Goniewicz, K., Borowska-Stefańska, M., Wiśniewski, S. and Dulebenets, M. A. (2022) 'Uncertainties in liner shipping and ship schedule recovery: A state-of-the-art review', *Journal of Marine Science and Engineering*, 10(5), p. 563.
- Forrest, S. (1996) 'Genetic algorithms', *ACM Computing Surveys (CSUR)*, 28(1), pp. 77-80.
- Gandomi, A. H., Yang, X. S. and Alavi, A. H. (2013) 'Cuckoo Search algorithm: a metaheuristic approach to solve structural optimization problems', *Engineering with Computers*, 29, pp. 17-35.
- Gao, Y., Chen, C. H. and Chang, D. (2023) 'A machine learning-based approach for multi-agv dispatching at automated container terminals', *Journal of Marine Science and Engineering*, 11(7), p. 1407.
- Giuffrida, N., Stojaković, M., Twrdy, E. and Ignaccolo, M. (2021) 'The Importance of environmental factors in the planning of container terminals: The case study of the port of Augusta', *Applied Sciences*, 11(5), p. 2153.
- Holland, J. H. (1992) 'Genetic algorithms', *Scientific American*, 267(1), pp. 66-73.
- Jugović, A., Sirotić, M. and Peronja, I. (2021) 'Sustainable development of port cities from the perspective of transition management', *Transactions on Maritime Science*, 10(2), pp. 466-476.
- Kim, K. H. and Park, Y. M. (2004) 'A crane scheduling method for port container terminals', *European Journal of Operational Research*, 156(3), pp. 752-768.
- Kovalenko, A. (2024) 'Modelling container ship transport flow: an application to alternative sea routes between Northeast Asia and Northwest Europe', *Maritime Policy & Management*, 51(4), pp. 572-587.
- Li, H. and Li, X. (2022) 'A branch-and-bound algorithm for the bi-objective quay crane scheduling problem based on efficiency and energy', *Mathematics*, 10(24), p. 4705.
- Miller, B. L. and Goldberg, D. E. (1995) 'Genetic algorithms, tournament selection, and the effects of noise', *Complex Systems*, 9(3), pp. 193-212.
- Moccia, L., Cordeau, J. F., Gaudioso, M. and Laporte, G. (2006) 'A branch-and-cut algorithm for the quay crane scheduling problem in a container terminal', *Naval Research Logistics (NRL)*, 53(1), pp. 45-59.
- Msakni, M. K., Al-Salem, M., Rabadi, G., Kotachi, M. and Diabat, A. (2018) 'Quay crane scheduling with vessel stability', *Transportation Research Procedia*, 30, pp. 60-69.
- Ngoc, C. T., Xu, X., Kim, H. S., Nguyen, D. A. and You, S. S. (2022) 'Container port throughput analysis and active management using control theory', *Proceedings of the Institution of Mechanical Engineers, Part M: Journal of Engineering for the Maritime Environment*, 236(1), pp. 185-195.
- Riadi, A., Putra, G. L. and Budiyo, M. A. (2023) 'Research on the effect of number of yard trucks on container terminal throughput', *Maritime Technology and Research*, 5(3), 262975.
- Rodrigues, F. and Agra, A. (2024) 'Handling uncertainty in the quay crane scheduling problem: a unified distributionally robust decision model', *International Transactions in Operational Research*, 31(2), pp. 721-748.
- Ursavas, E. (2017) 'Crane allocation with stability considerations', *Maritime Economics & Logistics*, 19, pp. 379-401.
- Van Twiller, J., Sivertsen, A., Pacino, D. and Jensen, R. M. (2024) 'Literature survey on the container stowage planning problem', *European Journal of Operational Research*, 317(3), pp. 841-857.
- Wang, J., Hu, H. and Song, Y. (2013) 'Optimization of quay crane scheduling constrained by stability of vessels', *Transportation Research Record*, 2330(1), pp. 47-54.

- Wang, Y. and Kim, K. H. (2011) 'A quay crane scheduling algorithm considering the workload of yard cranes in a container yard', *Journal of Intelligent Manufacturing*, 22, pp. 459-470.
- Wilson, J., Carriveau, R., Hurley, W., Babaei, R. and Ting, D. S. (2024) 'Electrifying Maritime Shipping: Evaluating the CO2 Reduction and Viability of Future Battery-Powered Container Ships Calling at the Port of Los Angeles', in *Journal of Physics: Conference Series*, Vol. 2929, No. 1. IOP Publishing, 012007.
- Yachba, K., Roufaida, L. and El Batoul, B. F. (2022) 'Cuckoo Search for the Resolution of the Container Storage Problem', *International Journal of Organizational and Collective Intelligence (IJOI)*, 12(1), pp. 1-14.
- Yang, X. S. (2009) 'Firefly algorithms for multimodal optimization', in *International Symposium on Stochastic Algorithms*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 169-178.
- You, S. S., Cuong, T. N. and Kim, H. S. (2025) 'Optimizing quay crane scheduling using deep reinforcement learning with hybrid metaheuristic algorithm', *Engineering Applications of Artificial Intelligence*, 143, 110021.
- Zhang, Q., Zhu, Y., Qin, J., Duan, J., Zhou, Y., Shi, H. and Nie, L. (2025) 'Study on the Multi-Equipment Integrated Scheduling Problem of a U-Shaped Automated Container Terminal Based on Graph Neural Network and Deep Reinforcement Learning', *Journal of Marine Science and Engineering*, 13(2), p. 197.
- Zhang, Z., Liu, M., Lee, C. Y. and Wang, J. (2018) 'The quay crane scheduling problem with stability constraints', *IEEE Transactions on Automation Science and Engineering*, 15(3), pp. 1399-1412. Available at: <https://doi.org/10.1109/TASE.2018.2795254>
- Zheng, Y., Xu, M., Wang, Z. and Xiao, Y. (2023) 'A genetic algorithm for integrated scheduling of container handling systems at container terminals from a low-carbon operations perspective', *Sustainability*, 15(7), p. 6035.