Comparing Encoder Architectures for the Discrete Berth Allocation Problem

Testing a pure attention transformer mechanism as an alternative to the GAT encoder a Reinforcement Learning agent uses to solve the Discrete Berth Allocation Problem

Author: Codrin Radetchi **Supervisor(s): Carlos March Moya, Neil Yorke-Smith**

[1]EEMCS, Delft University of Technology, The Netherlands

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
in Partial Fulfilment of the Requirements
for the Bachelor of Computer Science and Engineering
15th June, 2026

Name of the student: Codrin Radetchi
Final project course: CSE3000 Research Project
Thesis committee: Neil Yorke-Smith, Carlos March Moya, Wendelin Böhmer

Comparing Encoder Architectures for the Discrete Berth Allocation Problem

Author: Codrin Radetchi

Supervisors: Carlos March Moya, Neil Yorke-Smith

Abstract

This study looks at whether we can replace the Graph Neural Network (GNN) encoder in a reinforcement learning framework with a direct attention mechanism to solve the Dynamic Discrete Berth Allocation Problem (DDBAP): the challenge of assigning ships to specific docking spots over time. We tested two alternative designs, a Pure Attention Transformer and a topology-aware Edge-Transformer, against a baseline Graph Attention Network (GAT) using 27 simulated shipping scenarios. On average, the baseline GNN performed relatively better overall, scoring an average cost of 646.58 compared to 650.72 for the Edge-Transformer and 658.63 for the Pure Transformer. However, the best choice depends entirely on the size of the problem. The GNN works best in low-traffic situations with plenty of available berths because it ignores irrelevant background information. In contrast, when the port gets highly congested, such as 120 ships competing for just 5 berths, the global attention mechanism performs much better because it can anticipate long-term queue delays. Finally, while the attention-based models take significantly longer to train over 30,000 samples, they both process decisions in less than a second during live testing. This makes them highly practical for real-time maritime scheduling.

1 Introduction

The efficiency of maritime logistics serves as a fundamental pillar for global supply chains, yet port operations remain a significant bottleneck in the transport of goods. Within this domain, the Berth Allocation Problem (BAP) [6] is a critical operational challenge, requiring the assignment of arriving vessels to specific berths at optimal times to minimize total service time. When these vessels arrive dynamically over a rolling horizon, a scenario known as the Dynamic Discrete Berth Allocation Problem (DDBAP) [12], the complexity increases significantly due to real-time uncertainty and the NP-hard [7] nature of the underlying optimization. Beyond operational efficiency, improving berth allocation has profound environmental implications; for instance, emissions from port calls and anchorage waiting can reach levels comparable to the annual emissions of entire countries.[1, 8, 3]

Traditional approaches to the DDBAP often rely on exact Mixed-Integer Programming (MIP) [11], genetic algorithms [4, 2, 14] or metaheuristics [10]. While exact solvers can find optimal solutions for small instances, they fail to scale for real-time port operations, which frequently require immediate re-optimization. Conversely, simple dispatching rules like First-Come-First-Served (FCFS) are computationally efficient but often result in large optimality gaps because they ignore the global scheduling structure. Recent research has shifted toward Reinforcement Learning (RL), which allows agents to learn decision-making policies from experience. A notable advancement by March Moya et al. introduced a size-agnostic RL framework [5] using Graph Neural Networks (GNNs) to encode the port state as a heterogeneous bipartite graph [13]. While GNNs are effective at capturing the

spatial relationships between vessels and berths, the reliance on graph-based message passing introduces specific architectural complexities and computational overhead.

Despite the success of GNN-based encoders in maritime logistics, the potential of using direct attention mechanisms popularized by transformer architectures as a replacement remains an open question in the context of DDBAP. Attention mechanisms can model global dependencies without the explicit requirement of graph edges, potentially offering a more flexible or efficient state representation [9]. However, it is unclear whether removing the inductive bias of the graph structure in favour of direct attention maintains the necessary performance levels for complex scheduling.

The primary research question of this study is: **Can the encoder of the DDBAP methods be replaced by a direct attention mechanism instead of a Graph Neural Network (GNN)?** Specifically, this research investigates how this architectural shift affects the agent's performance in terms of the optimality gap and the associated computational cost. We will compare the new proposed architecture with the GNN over 27 synthetically generated instances. Moreover, we will also use the following four heuristic approaches to the DDBAP as a benchmark for our architecture's performance: **FCFS** (assigns the earliest-arrived vessel to the available berth with the shortest processing time), **SPT**(always selects the vessel-berth pair with the shortest processing time), **Priority** (dispatches vessels in decreasing priority order, assigning each to the available berth with the shortest processing time), and **WTSP** (ranks all arrived-vessels/free-berths pairs by the ratio of the vessel's priority to the pair's processing time and selects the highest scoring pair). WTSP is used as a reference solution for reward scaling.

This paper contributes to the field by implementing an attention-based encoder for the DDBAP agent described by March Moya et al. and providing a rigorous comparative analysis. The rest of this paper is structured as follows: Section 2 describes the DDBAP formulation and the baseline RL methodology. Section 3 details the proposed attention-based encoder architecture. Section 4 presents the experimental setup and results, comparing the performance against GNN-based agents and classical heuristics. Section 5 discusses the ethical and reproducibility aspects of the research. Finally, Section 6 and Section 7 provide a discussion of the findings and concluding remarks.

2 Problem Description and Background

2.1 The Dynamic Discrete Berth Allocation Problem (DDBAP)

The Berth Allocation Problem (BAP) is a combinatorial optimization challenge central to maritime logistics, concerning the spatial and temporal assignment of arriving vessels to available quayside berths. This study addresses the **Dynamic and Discrete** variant (DDBAP). The problem is discrete because the quay is partitioned into a finite set of specific, fixed berthing positions B, where each berth $b \in B$ can service at most one vessel at any given time. It is dynamic because vessels arrive continuously over an infinite or rolling time horizon, rather than being present at the port simultaneously at time zero.

Formally, a set of vessels V must be scheduled. Each vessel $v \in V$ is characterized by a tuple:

$$\mathcal{V}_v = (a_v, p_v, c_v) \tag{1}$$

where a_v represents its expected arrival time, p_v is its handling time (which depends on the vessel's cargo volume and the assigned berth's crane capabilities), and c_v represents its marginal delay cost or priority weight. The objective of the DDBAP is to find an assignment

policy that maps each vessel to a berth and a precise handling start time $s_v \geq a_v$ such that total weighted service time, encompassing both waiting time and handling duration, is minimized, while strictly respecting spatial boundaries and temporal non-overlapping constraints.

2.2 Operational Uncertainty and Information Horizons

In practical maritime operations, solving the DDBAP via traditional offline optimization methods (e.g., Mixed-Integer Linear Programming or classical metaheuristics) presents severe limitations due to two compounding factors:

- **Real-Time Uncertainty:** The arrival time a_v and handling duration p_v are inherently stochastic. Weather disruptions, sea state variability, mechanical failures, and labor delays mean that static schedules become sub-optimal or completely infeasible shortly after execution begins.
- **Limited Information Horizon:** Ports do not possess perfect, omniscient information regarding the entire fleet V at the start of the planning horizon. Instead, information is revealed incrementally. A port authority operates within a restricted visibility window, discovering precise vessel metrics only when a ship enters a certain radar radius or submits a final notice of arrival.

Consequently, the port must treat berth allocation not as a static, one-shot scheduling problem, but as a sequential, dynamic decision-making process under uncertainty.

2.3 The MDP Formulation for Berth Allocation

To effectively capture this sequential nature, recent advancements reformulate the DDBAP as a **Markov Decision Process (MDP)**. The MDP framework is uniquely suited here because it models an environment where an agent takes actions based on incomplete, evolving states, directly optimizing for long-term operational efficiency rather than immediate greedy gains.

An MDP is formally defined by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$, which maps directly to the maritime operational environment:

- **State Space ($\mathcal{S}$):** The state $s_t \in \mathcal{S}$ represents a comprehensive snapshot of the port at decision epoch t. This includes the current occupancy and remaining handling times of all berths, alongside the attributes (waiting times, priority scores) of unassigned vessels currently in the anchorage queue.
- **Action Space ($\mathcal{A}$):** An action $a_t \in \mathcal{A}$ consists of assigning an available vessel from the queue to an idle berth. If no optimal assignment can be made, or if it is strategically advantageous to wait for a high-priority vessel that is arriving shortly, the agent can issue a "no-action" or hold command.
- **Transition Probability ($\mathcal{P}$):** This governs how the port state evolves from s_t to s_{t+1} after an assignment. It implicitly captures the environment's environmental stochasticity, such as random delays in vessel arrivals or unexpected extensions in crane processing times.

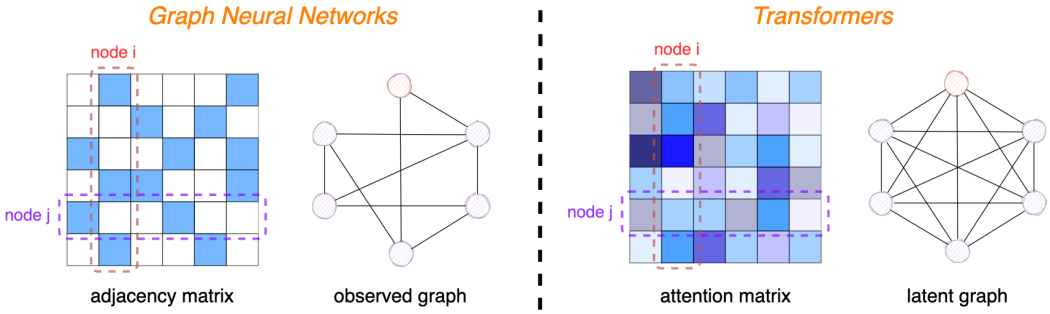
- **Reward Function ($\mathcal{R}$):** The reward r_t acts as the feedback mechanism driving policy convergence. To minimize total service time, the reward is typically structured as a negative penalty proportional to the accumulation of vessel waiting costs and berth idle times incurred during the step.

By framing the DDBAP as an MDP, Reinforcement Learning (RL) agents can learn robust, reactive policies. Rather than re-solving an entire NP-hard optimization problem from scratch whenever an operational delay occurs, the trained RL agent evaluates the state space in real time, making immediate, computationally negligible inference decisions that remain structurally sound under high variance.

3 Proposed encoder alternatives

3.1 Pure Attention Transformer

As explained before, our transformer computes global attention (each node's embedding will be influenced by all other nodes), unlike GAT which considers only pairs of nodes connected by an edge.



Inherently, it uses a different formula to calculate the attention score of two nodes:

$$e_{ij} = \frac{\mathbf{q}_i \mathbf{k}_j^T}{\sqrt{d_k}}$$

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{m=1}^{N} \exp(e_{im})}$$

Where:

$\mathbf{q}_i$ Query vector for token i, derived from the embedding ($\mathbf{x}_i W_Q$).

$\mathbf{k}_j^T$ Transposed key vector for token j, derived from the embedding ($W_K^T \mathbf{x}_j^T$).

d_k Dimensionality of the query/key vectors (used as a scaling factor).

e_{ij} Raw, unnormalized attention score between token i and token j.

N Total sequence length (total number of tokens).

α_{ij} Final attention weight (probability) token i assigns to token j calculated by applying the softmax function to e_{ij}.

$\exp(x)$ The exponential function, evaluated as e^x where e is Euler's number.

3.2 Edge-biased transformer

While homogenous (vessel-to-vessel or berth-to-berth) relationships are relevant, edge related nodes hold significant contextual information that our agent should be incentivized to pay more attention to. Therefore, one approach to improve the previously studied transformer architecture would be to introduce a slight bias towards nodes connected by an edge, essentially getting closer to the GAT philosophy, but still keeping all nodes as context and utilizing the transformer attention formula [15]. We do this by constructing a mask which holds a positive scalar for each vessel-berth pair and 0 otherwise, and adding it to our attention matrix before applying softmax. In other words, we are artificially increasing the attention score of neighbour nodes. Each scalar value is derived from that specific edge's embedding by transposing its feature vector into h dimensions, where h is the number of attention heads, and h_i is the edge score (scalar) that we add to the nodes' attention score in the i-th attention head.

Therefore, the attention score in the i-th attention head is formulated as:

$$
\begin{aligned}
e_{ij}^{(i)} &= \frac{\mathbf{q}_i \mathbf{k}_j^T}{\sqrt{d_k}} + \mathbf{B}_{ij}^{(i)} \\
\alpha_{ij}^{(i)} &= \frac{\exp(e_{ij}^{(i)})}{\sum_{m=1}^{N} \exp(e_{im}^{(i)})}
\end{aligned}
$$

Where:

$\mathbf{q}_i, \mathbf{k}_j^T$ Query and transposed key vectors derived from the node embeddings.

d_k Dimensionality of the query/key vectors.

h_i The scalar edge score for head i, extracted from the i-th dimension of the projected edge embedding vector $\mathbf{h} \in R^H$ (where H is the total number of attention heads).

$\exp(x)$ The exponential function, evaluated as e^x where e is Euler's number.

$\alpha_{ij}^{(i)}$ The final, topology-aware attention weight in head i.

$\mathbf{B}_{ij}^{(i)}$ The heterogeneous bias mask for attention head i, defined as:

$$
\mathbf{B}_{ij}^{(i)} = \begin{cases} 0 & \text{if token } i \text{ and token } j \text{ are homogeneous} \\ & \text{(vessel-vessel / berth-berth),} \\ h_i & \text{if token } i \text{ and token } j \text{ are heterogeneous} \\ & \text{and share an edge.} \end{cases}
$$

4 Experimental Setup and Results

To systematically evaluate the performance of different encoder architectures within this framework, a controlled synthetic testing suite was generated. The instance generation process assumes exponential inter-arrival times for vessels to accurately simulate real-world maritime traffic queuing. A dedicated evaluation suite was constructed consisting of 27 distinct instances. These instances systematically vary the number of vessels ($V \in \{80, 100, 120\}$), the number of berths ($B \in \{5, 10, 15\}$), and a congestion multiplier ($C \in \{0.5, 1.0, 2.0\}$), providing a comprehensive spectrum of operational pressures ranging from sparse traffic to severe bottleneck scenarios.

To provide an objective performance benchmark, a separate validation suite consisting of 6 distinct instances was constructed. Within this controlled testing environment, the proposed model was evaluated alongside the following four heuristic approaches: **FCFS** (assigns the earliest-arrived vessel to the available berth with the shortest processing time), **SPT**(always selects the vessel-berth pair with the shortest processing time), **Priority** (dispatches vessels in decreasing priority order, assigning each to the available berth with the shortest processing time), and **WTSP** (ranks all arrived-vessels/free-berths pairs by the ratio of the vessel's priority to the pair's processing time and selects the highest scoring pair). WTSP is used as a reference solution for reward scaling.

The proposed pure transformer agent and edge-transformer agent were both trained on 30,000 samples to ensure convergence. During evaluation, both agents acted deterministically to schedule all vessels. The primary metric of comparison is the normalized objective value, representing the total weighted waiting and handling times across the scheduling horizon. Lower scores indicate superior scheduling policies.

4.1 Pure Transformer's Results

4.1.1 Validation Set Performance

Table 1: **Validation Set Performance: Heuristics vs. Agent**

Instance	Config	FCFS	SPT	Priority	WTSP	Agent	Winner
instance_1	10v / 3b / C0.5	724.40	378.10	387.60	334.50	**322.30**	**Agent**
instance_2	10v / 3b / C1.0	1137.30	947.70	1315.90	904.50	**592.80**	**Agent**
instance_3	10v / 3b / C2.0	768.40	643.80	718.60	494.70	**449.40**	**Agent**
instance_4	30v / 5b / C0.5	1134.37	524.40	953.23	555.10	**366.00**	**Agent**
instance_5	30v / 5b / C1.0	1554.47	741.57	1416.97	692.47	**418.27**	**Agent**
instance_6	30v / 5b / C2.0	1092.03	544.93	972.37	550.30	**492.00**	**Agent**
Average		**1068.50**	**630.08**	**960.78**	**588.60**	**440.13**	**Agent (6/6 wins)**

4.1.2 Test Set Performance (27 instances)

Average Normalized Cost: GNN Agent: '646.5799'; Transformer Agent: '658.6327'
Overall Difference: The Transformer Agent is on average '+1.86
Win/Loss Record: 6 wins for the Transformer Agent, 21 losses for the GNN Agent.

4.2 Edge-Transformer's Results

4.2.1 Validation Set Performance

Table 2: Validation Set Performance: Heuristics vs. Agent

Instance	Config	FCFS	SPT	Priority	WTSP	Agent	Winner
1	10v / 3b / C0.5	724.40	378.10	387.60	334.50	334.50	Tie
2	10v / 3b / C1.0	1137.30	947.70	1315.90	904.50	**530.40**	**Agent**
3	10v / 3b / C2.0	768.40	643.80	718.60	494.70	494.70	Tie
4	30v / 5b / C0.5	1134.37	524.40	953.23	555.10	**340.63**	**Agent**
5	30v / 5b / C1.0	1554.47	741.57	1416.97	692.47	**417.87**	**Agent**
6	30v / 5b / C2.0	1092.03	544.93	972.37	550.30	**434.63**	**Agent**
Average		**1068.50**	**630.08**	**960.78**	**588.60**	**425.46**	**Agent (4 wins, 2 ties)**

4.2.2 Test Set Performance

Average Normalized Cost: GNN Agent: `646.5799`; Edge-Transformer Agent: `650.7153`
Overall Difference: The Edge-Transformer Agent is on average `+0.64%` worse than the GNN Agent.
Win/Loss Record: 9 wins for the Edge-Transformer Agent, 18 wins for the GNN Agent.

5 Responsible Research

The methodology prioritizes reproducibility and computational transparency. All test instances were generated using fixed random seeds to ensure that the exact arrival distributions and handling times can be consistently replicated by future researchers. Furthermore, relying on synthetically generated exponential arrival data bypasses the privacy and confidentiality constraints often associated with proprietary port manifests. Ethically, advancing DDBAP optimization carries significant environmental implications. Vessels anchored offshore waiting for a berth burn substantial amounts of fuel, contributing heavily to maritime emissions. By developing more efficient, scalable scheduling algorithms, this research directly supports global efforts to reduce the carbon footprint of maritime logistics.

6 Discussion

While the aggregate results suggest a slight overall advantage for the GNN encoder, segmenting the instances by congestion uncovers a distinct architectural trade-off. In low-congestion scenarios (configurations with 15 berths), the problem behaves highly locally. The primary challenge is matching a specific vessel to its most compatible berth without severe queueing considerations. Under these conditions, the GNN dominates, winning 8 out of 9 instances. The GNN's inductive bias, restricting message passing only to valid local connections, is perfectly suited for this, whereas the Transformer's global attention introduces unnecessary noise by evaluating every possible global connection path. Conversely, under extreme congestion (configurations with 5 berths) and increasing problem scale, the Edge-Transformer demonstrates a significant advantage. As the problem scales to 100 and 120 vessels competing for only 5 berths, the scheduling challenge shifts from local matching to global queue

optimization. In the 120-vessel, 5-berth category, the Transformer won all three instances, including a large performance improvement on the high-congestion $C = 2.0$ instance (reducing the objective cost by 77.40 points). The global attention mechanism allows the agent to formulate long-range scheduling paths and anticipate queue dynamics far more effectively than a GNN, which requires multiple local hops to propagate the same global state information.

An intriguing structural anomaly emerges when analyzing the validation set performance: the pure attention Transformer wins cleanly across all 6 baseline instances (6/6 wins), whereas the structurally superior Edge-Transformer records two outright ties (4 wins, 2 ties) against the WTSP heuristic. This initial friction is directly attributable to the strict parameters of the validation suite, which consists exclusively of small-scale, 10-vessel and 30-vessel instances. In these highly constrained, small state spaces, the mathematical behavior of the WTSP heuristic frequently hits the exact theoretical optimum. Because the global Transformer optimizes unhindered by topological restrictions, it maps wide-horizon routing combinations that match these absolute lower bounds perfectly. Conversely, the Edge-Transformer introduces an artificial topology-aware bias by adding projected edge embeddings to its raw attention matrix prior to the softmax calculation. On tiny validation graphs, this inductive bias forces the network to over-prioritize immediate vessel-berth pairings, restricting its exploration path and causing it to converge identically to the local greedy logic of WTSP. However, this exact restriction is precisely what enables the Edge-Transformer to outperform the pure Transformer on the expansive 27-instance test suite (650.72 vs. 658.63 average cost). As the environment scales up to 120 vessels, the unconstrained global Transformer encounters significant noise by propagating attention across irrelevant distal nodes. In these intense bottleneck distributions, the Edge-Transformer's built-in structural bias acts as an essential regularizer, preventing the global attention from dissolving into chaos while still preserving enough wide-horizon sight to capture complex queue dynamics.

Beyond scheduling performance, a critical axis of comparison between the two architectures is their underlying computational complexity and its implications for real-time deployment. The baseline Graph Attention Network encoder operates with a computational complexity scaling linearly with respect to the number of edges in the bipartite graph, restricting its attention computation exclusively to immediate topological neighbors. In contrast, the global attention mechanism of the Transformer incurs a quadratic computational complexity ($O(N^2)$) relative to the total sequence length N, as it evaluates every pairwise combination of tokens regardless of graph connectivity. During training, this architectural difference manifested in a substantial computational overhead for both Transformer variants, demanding longer wall-clock execution times to process the same number of training instances. However, during deterministic evaluation, this disparity narrowed significantly. Because real-time re-optimization requires only a forward pass of the network, both encoders executed within sub-second operational windows. This indicates that while the Edge-Transformer demands vastly higher computational resources during its 30,000-sample training phase to achieve convergence, its inference footprint remains highly viable for live maritime dispatching environments.

Furthermore, the structural efficiency of the models is reflected in their respective parameter count, where the two proposed transformer architectures demonstrate a highly compact representation. The baseline GNN encoder requires the largest model capacity at 170,820 parameters, a number driven by the multi-layer message-passing operations and localized state aggregation networks required to preserve graph topology. In contrast, both proposed

encoders significantly compress this figure; the pure transformer operates with 122,180 parameters, while the edge-transformer requires 122,208 parameters, representing a roughly 28.5% decrease in total parameter scale relative to the GNN baseline. This parameter reduction highlights a key architectural advantage: by replacing explicit, deep graph-convoluted message passing with unified self-attention mechanisms, the proposed encoders achieve superior high-congestion scaling and competitive low-congestion performance while relying on a substantially leaner parameter space. This lean footprint minimizes memory bandwidth requirements and training computational cost, further enhancing the real-time deployment viability of the proposed frameworks.

7 Conclusions and Future Work

In this study, we evaluated whether replacing a Graph Neural Network encoder with a direct attention mechanism could improve a Reinforcement Learning framework for the Dynamic Discrete Berth Allocation Problem. By testing these models across 27 simulated port scenarios, we discovered a clear trade-off between the local focus of GNNs and the broad overview provided by attention mechanisms.

On average across all tests, the baseline GNN performed slightly better overall, scoring an average normalized cost of 646.58, compared to 650.72 for the Edge-Transformer and 658.63 for the Pure Attention Transformer. However, this overall average hides a major shift in performance caused by the scale and traffic of the port:

- **Low-Congestion Scenarios:** When traffic is light and berths are readily available, the GNN performs best. Its structural design naturally filters out irrelevant background data, allowing it to excel at simple local assignments.
- **High-Congestion Bottlenecks:** When the port gets highly crowded, such as 120 ships competing for just 5 berths, the attention mechanism's global perspective becomes highly superior. In these intense scenarios, the Transformer agents consistently beat the GNN because they are better at anticipating long-term queue patterns and reducing total delays.

Our computational analysis also showed a clear trade-off in efficiency. Although the Transformer models take significantly longer to train over 30,000 samples to reach peak performance, making real-time scheduling decisions takes less than a second. This sub-second processing speed proves that attention-based models are highly practical for live maritime dispatch systems.

To address the fact that different models win at different scales, future research should explore hybrid models that adapt to changing port conditions. One highly promising approach is to build a dynamic routing agent or switching mechanism. This system would automatically use local GNN message-passing when port traffic is sparse, and then seamlessly switch to a global Transformer attention layer as congestion increases toward critical levels. Additionally, adding time-based (temporal) encoding directly into the ship tokens could help the agent better predict future arrival patterns without depending entirely on secondary edge features.

References

[1] Yasuhiro Akakura. Analysis of offshore waiting at world container terminals and estimation of co2 emissions from waiting ships. *Asian Transport Studies*, 9:100111, 2023. doi:10.1016/j.eastsj.2023.100111.

[2] Carlos Arango, Pablo CortÃ©s, Alejandro Escudero, and Luis Onieva. 17 - genetic algorithm for the dynamic berth allocation problem in real time. In Xin-She Yang, Zhihua Cui, Renbin Xiao, Amir Hossein Gandomi, and Mehmet Karamanoglu, editors, *Swarm Intelligence and Bio-Inspired Computation*, pages 367–383. Elsevier, Oxford, 2013. doi:10.1016/B978-0-12-405163-8.00017-X.

[3] Blue Visby Consortium. Overcoming 'sail fast, then wait': The carbon reduction potential of global maritime optimization. White paper, Blue Visby Consortium, 2022. Available at https://bluevisby.com/.

[4] Shu-Chuan Chang, Ming-Hua Lin, and Jung-Fa Tsai. An optimization approach to berth allocation problems. *Mathematics*, 12(5), 2024. doi:10.3390/math12050753.

[5] Majid Ghasemi and Dariush Ebrahimi. Introduction to Reinforcement Learning, 2024. arXiv:2408.07712, doi:10.48550/arXiv.2408.07712.

[6] Mihalis Golias, Georgios Saharidis, M. Boile, S. Theofanis, and Marianthi Ierapetritou. The berth allocation problem: Optimizing vessel arrival time. *Maritime Economics and Logistics*, 11:358–377, 2009. doi:10.1057/mel.2009.12.

[7] Mahdi Imanparast. Challenges and strategies for tackling NP-hard problems. In *9th International Conference on Combinatorics, Cryptography, Computer Science and Computation*, 2024. URL: https://www.researchgate.net/publication/387577142_Challenges_and_strategies_for_tackling_NP-hard_problems.

[8] International Maritime Organization. *Fourth IMO Greenhouse Gas Study 2020*. International Maritime Organization, London, UK, 2020. Available at https://www.imo.org/.

[9] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. 2020. URL: https://arxiv.org/abs/2001.08361, arXiv:2001.08361.

[10] NataÅ¡a KovaÄ. Metaheuristic approaches for the Berth Allocation Problem. *Yugoslav Journal of Operations Research*, 27:265–289, 2017. doi:10.2298/YJOR160518001K.

[11] Sirui Li, Janardhan Kulkarni, Ishai Menache, Cathy Wu, and Beibin Li. Towards foundation models for mixed integer linear programming, 2025. URL: https://arxiv.org/abs/2410.08288, arXiv:2410.08288.

[12] Shih-Wei Lin, Kuo-Ching Ying, and Shu-Yen Wan. Minimizing the total service time of discrete dynamic berth allocation problem by an iterated greedy heuristic. *TheScientificWorldJournal*, 2014:218925, 2014. doi:10.1155/2014/218925.

[13] Carlos March Moya, Frederik Schulte, Kevin Tierney, and Neil Yorke-Smith. A reinforcement learning approach for the dynamic berth allocation problem. In *Proceedings of LOGMS 2026*, 2026.

[14] Sima Sargazi, Hassan Mishmast Nehi, Hamed Ahmadzade, and Jafare Sayareh. Optimizing berth allocation problem with irregular layout using an improved genetic algorithm under uncertain conditions. *Fuzzy Information and Engineering*, 18(1):43–73, 2026. doi:10.26599/FIE.2026.9270004.

[15] Chao Wang, Jiaxuan Zhao, Lingling Li, Licheng Jiao, Fang Liu, and Shuyuan Yang. Automatic graph topology-aware transformer. *arXiv preprint arXiv:2405.19779*, 2024. URL: https://arxiv.org/abs/2405.19779, arXiv:2405.19779.