Reinforcement Learning for the Discrete Dynamic Berth Allocation Problem

Evaluating a trained Discrete Dynamic Berth Allocation model on Berth breakdowns

Tautvydas Kuklys[1]
Supervisor(s): Carlos March Moya, Neil Yorke-Smith[1]

[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
June 21, 2026

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

Reinforcement Learning for the Discrete Dynamic Berth Allocation Problem
Evaluating a trained Discrete Dynamic Berth Allocation model on Berth breakdowns

Tautvydas Kuklys

EEMCS, Delft University of Technology, The Netherlands

Supervisor(s): Carlos March Moya, Neil Yorke-Smith

Thesis committee: Neil Yorke-Smith, Carlos March Moya, Wendelin Böhmer

Abstract

The problem of scheduling vessels in a port as they arrive one-by-one is known as the Dynamic Berth Allocation Problem and it is NP-hard. This paper analyses the influence of berth breakdowns on the scheduling and optimality of a trained Reinforcement Learning model by March Moya et al. [9] for such a problem. Several breakdown parameters, including frequency, severity, duration, and the probability of binary versus partial breakdowns, were examined independently and in combination with one another with respect to different scheduling heuristics. The breakdowns were dynamically injected into the model's event loop so that it did not have knowledge of upcoming breakdowns. Each experimental configuration was evaluated using ten random seeds and the sample mean and standard deviation were computed. The results showed low variance between different seeds and configurations. Breakdown frequency was the main factor limiting the model's performance, moving the performance from a 19.6% advantage to 6.4% in the most extreme cases when compared to the baseline heuristic of WTSP. The other parameters did not produce significant model degradation.

Keywords: *Dynamic berth allocation problem, Reinforcement Learning, Binary and partial berth breakdowns, Breakdown severity*

1 Introduction

Dynamic berth allocation to vessels is one of the most prominent challenges in maritime logistics: not only does efficient and orderly assignment of berths (docks) to vessels lower queue waiting times, but it also lowers carbon emissions. According to estimates from the UCL Shipping and Oceans Research Group, vessels spend a considerable amount of time waiting at ports, and converting this waiting into slower sailing could yield potential greenhouse gas emission reductions of approximately 10–25%, depending on vessel type [12]. Furthermore, Rotterdam's maritime supply chain emissions alone reach 13.7 Mt CO_2 — comparable to the output of major coal-fired power plants [13]. Therefore, optimizing berth allocation will both reduce operational costs and also be an important step in reducing our carbon emissions for a carbon-neutral future.

The problem of assigning berths to vessels with the aim of minimizing some objective value is known as the **Berth Allocation Problem (BAP)**. The variation of the problem where vessels are arriving dynamically and are not all present at the start of the docking period is known as the **Dynamic Berth Allocation Problem (DBAP)**. Indeed, the DBAP is *NP-hard*, making it extremely difficult for ports to efficiently and strategically assign vessels to berths in an orderly manner in real time. Simple heuristics like **first-come, first-served (FCFS)** can be used to greedily assign vessels to berths, but they do not optimally capture the global state.

Reinforcement learning (RL) is a part of machine learning concerned with sequential decision-making under uncertainty. Rather than optimizing greedily at each step, an RL agent learns a policy that maximizes long-term discounted cumulative reward. It takes into account the fact that the consequences of an action may only become apparent much later. This almost perfectly models the problem we are dealing with in the DBAP. Therefore, reinforcement learning can be used as a natural way to estimate and optimize berth and vessel allocations in real time.

Operational disruptions are a major part of port planning operations. They cannot be predicted but are an inescapable reality of any supply chain process. Inefficient handling of disruptions lowers the throughput of vessels, increases vessel turnaround time and creates congestion. This makes the ports unattractive for investors and vessels and has secondary effects like increased greenhouse gas emissions from vessels [12]. Therefore, agility is a key characteristic of an effective port. Successful ports adapt quickly, react to breakdowns and provide remedial actions for vessels.

Although learning-based methods provide a promising solution for the DBAP, ensuring fast sequential decision-making and capturing the global state of the port, their reliability under operational disruptions still remains insufficiently understood. In real port operations, berths can become unavailable due to equipment failure, breakdowns, accidents, weather disruptions or just regular maintenance. Such breakdowns can influence policy decisions: we now have to find another optimal schedule, optionally reassigning the vessel where the breakdown happened. A learned model can therefore experience degradation when it encounters data it has not been trained on, but which remains critical in real-life scenarios. This raises a question: does the trained model remain robust when it encounters operational disruptions?

Specifically, this paper is concerned with how the prob-

lem and its optimality with respect to a pre-trained model degrades as we introduce berth breakdowns. In particular, we will answer:

- **RQ**: How well does the Graph Neural Network (GNN) -based DBAP agent by March Moya et al. [9] perform under operational disruptions such as berth breakdowns?

With the subquestions of:

- **SQ1**: How does performance of the trained GNN degrade as frequency, duration, and severity of berth breakdowns increase?
- **SQ2**: Does the model exhibit threshold effects, where performance suddenly collapses beyond a certain disruption level?
- **SQ3**: Does the model disproportionately delay certain classes of vessels under disruptions?

The remainder of this paper is structured as follows. *Section 2* discusses related work on the DBAP, reinforcement learning approaches for berth allocation, and robustness under disruptions. *Section 3* describes the problem and breakdown model that we will use for our experiments. *Section 4* showcases the experimental setup, different heuristics that we will compare to and the evaluation process. *Section 5* presents the performance degradation analysis. *Section 6* talks about the measures we took to provide a transparent research process and results. *Section 7* discusses the robustness implications and limitations of the results and *Section 8* concludes the paper.

2 Related work

2.1 Dynamic berth allocation

Previous work for the DBAP includes Lv et al. [7] where the agent selects among predefined heuristics at each step to resolve conflicts. Jo and Moon [3] propose a multi-agent process for joint vessel selection, berth allocation, and crane assignment, while Li et al. [6] train a size-agnostic model via evolutionary strategies for the continuous BAP (the variation of the problem where berths are continuous). Exact solutions for the DBAP include a Lagrangian relaxation [2] and a Mixed-Integer Programming solution [1]. Bee colony optimizations [10] still require computations done per-instance. Only March Moya et al. [9] provides a learning-based approach fit for a wide variety of scenarios: it proposes a unified model with a graph-based size-agnostic state representation, explicit vessel priority modelling, and a skip action (which allows waiting for future vessels if it is more globally optimal), enabling robust generalization across varying port configurations. In comparison, Li et al. [6] lacks a skip action and explicit priority modelling, Jo and Moon [3] encode only a simplified state without vessel priorities, and Lv et al. [7] lacks a skip action and is not size agnostic.

2.2 Disruptions and robustness in berth allocation

Research on berth allocation uncertainty has predominantly focused on vessel-related uncertainty, including stochastic arrival times and handling durations. Comparatively little attention has been devoted to infrastructure-related disruptions such as berth closures, maintenance activities, equipment failures, and temporary capacity reductions. These disruptions are important because they directly affect berth availability and operational constraints. Existing methods typically address such events through rescheduling or re-optimization, while the robustness of machine-learning-based allocation policies remains largely unknown. Rodrigues and Agra's [11] survey of berth allocation and quay crane scheduling under uncertainty found that the literature is dominated by stochastic optimization, robust optimization, and recovery-based approaches. Several relevant approaches are discussed below.

Umang et al. [14] propose a reactive framework that continuously updates berth schedules as new information becomes available, allowing terminals to adapt to changing operational conditions. Lyu et al. [8] propose a reactive collaborative berth planning approach for disruption recovery, emphasizing their new mixed-integer programming model that re-optimizes after a berth breakdown. Similarly, Zheng et al. [15] study the integrated rescheduling problem of berth allocation and quay crane assignment under uncertainty. Their objective is to restore operational efficiency through optimization-based recovery.

What all of these approaches miss is an evaluation of how existing scheduling policies, such as trained RL agents, perform under operational disruptions. Rather than proposing a new recovery mechanism, this paper evaluates how the performance of a trained reinforcement learning policy degrades under varying frequencies, durations, and severities of berth breakdowns.

For this reason, the work of March Moya et al. [9] provides a perfect benchmark: the paper provides a reinforcement learning solution for the DBAP but its behaviour under berth breakdowns has not been investigated. Consequently, a gap exists between research on disruption recovery and research on the robustness of learning-based berth allocation policies.

3 Methodology

3.1 Problem description

The original problem description of the dynamic berth allocation problem is as follows:

The port has berths $\mathcal{B}$ and incoming vessels $\mathcal{V}$. Each vessel has an assigned priority p_v and a berth-specific handling time $h_{v,b}$. Vessels arrive at some predefined point in the future a_v which is known to the agent. Vessels can only be assigned to berths after their arrival at the port at time $t_v \geq a_v$. Decision-points where we can choose to either assign a vessel to a berth or skip are known as events. Events include either *vessel arrivals or a freeing of a berth after fully handling a vessel*. The binary assignment variable $x_{v,b}$ equals 1 *iff* the vessel v is assigned to berth b, and 0 otherwise. All of the berths are available at the start of the problem and either one or zero vessels can be assigned to a berth at any time. The objective is to minimize the priority-weighted handling times:

$$\min \sum_{v \in \mathcal{V}} p_v \left(t_v - a_v + \sum_{b \in \mathcal{B}} h_{v,b} x_{v,b} \right)$$

3.2 Markov decision process

The model described in March Moya et al. [9] formulates the DBAP as a **Markov decision process (MDP)** where decisions are made sequentially for each event. At each event the agent either chooses to assign a berth-vessel pair or skip and subsequently receives a reward for its action.

The state is represented as a heterogeneous bipartite graph $(\mathcal{B}, \mathcal{V}, E)$ where each vessel is connected to each berth. A **Graph Neural Network (GNN)** processes this graph by passing messages between neighbouring nodes, allowing the agent to learn a scheduling policy that is aware of the global port state.

The vessel nodes are described by a four-dimensional feature vector that includes: the normalized waiting time that is already accumulated, the normalized waiting time until arrival, the normalized priority of the vessel and the normalized best achievable score across all berths.

The berth nodes are described by a three-dimensional vector that includes: the normalized waiting time until the berth is free, the normalized best handling time across all vessels and an occupancy indicator.

The edges are described by a five-dimensional vector that includes: the normalized ranking score for this berth-vessel pair, the normalized waiting cost already accumulated for this vessel, the normalized handling time and a three-dimensional one-hot status encoding showing if the vessel has yet to arrive, was skipped or is currently being processed on this berth.

Vessel arrivals follow the methodology of Lv et al. [7]: inter-arrival times are sampled from an exponential distribution with rate $\lambda = 1/I$, where

$$I = C \cdot \sum_{v,b} \frac{h_{v,b}}{|\mathcal{B}||\mathcal{V}|^2}$$

and $C > 0$ controls congestion (smaller C = higher congestion). Handling times are drawn from $\mathcal{U}[10, 200]$ and priorities from $\mathcal{U}\{1...10\}$. We choose $C \in \{0.5, 1.0, 2.0\}$ to represent low, medium, and high congestion scenarios.

In this paper, we use the MDP as a basis for the robustness analysis. Since our goal is not to reformulate and optimize the model, but to analyse its robustness under operational stress, we do not modify the process and only summarize the main components here. For the complete mathematical specification of the problem, including the state representation, reward formulation, transition dynamics, and RL model architecture, please consult March Moya et al. [9]. We will now discuss how we modelled and inserted the breakdowns into the current model and event-loop.

3.3 Terminology and assumptions

This subsection introduces the terminology and assumptions made when modelling the breakdowns. Some non-standard terminology that we will use:

- **Binary breakdowns**: breakdowns when the broken down berth is fully non-operational for the breakdown duration.
- **Partial breakdowns**: breakdowns where the broken down berth has increased handling times for all vessels.
- X^{th} **quantile breakdown duration** (D_x): a duration parameter that means that X% of breakdowns will have ended by that point.

Before we start modelling, we define several key scheduling assumptions regarding the scheduling of vessels:

- **Reactivity:** the model has no knowledge of future breakdowns, as opposed to vessel arrival times, which are fully known. Planned maintenance can be modelled as a ghost vessel with infinite priority, a known handling time and a reversible cost.
- **Occupancy-independent breakdowns:** breakdowns can also occur when there are no vessels on the berth. For instance, preventative maintenance needs to be done.
- **Breakdowns do not erase progress:** after a breakdown, only the proportional handling time already served on the current berth counts towards the objective - the vessel is not charged the full $h_{v,b}$. In addition, handling times are proportionally reduced for all of the other berths to reflect the work already completed.
- **Disruption response flexibility:** breakdowns create a new decision point event and vessels from the broken down berth are put into the waiting vessel queue. This is in contrast to the original model where vessels were only assigned to a berth once and processed fully. This was done to model flexible assignment when ports let vessels choose another berth due to unforeseen circumstances.

3.4 Breakdown Modelling

This subsection describes how berth breakdowns are introduced into the DBAP environment. We define when breakdowns occur, how long they last, and how they affect berth availability during the simulation.

Firstly, we need to decide whether to pick a binary or a partial breakdown event. The binary-partial split of breakdowns is taken from a Bernoulli trial:

$$\text{partial} \sim \text{Bernoulli}(p_{\text{partial}})$$

Breakdown events are generated according to a Poisson process with rate λ. Rather than sampling the total number of events directly, we generate the event start times sequentially using exponentially distributed inter-arrival times. If S_i denotes the start time of the i-th breakdown event, then

$$s_i = s_{i-1} + \Delta_i,$$

where

$$\Delta_i \sim \text{Exponential}(\lambda).$$

Here, Δ_i is the time between two consecutive breakdown events. The process starts at

$$s_0 = 0,$$

and new events are generated until the next sampled start time exceeds the simulation horizon T:

$$s_i < T.$$

For each generated event, the affected berth is selected uniformly at random from the set of berths:

$$b_i \sim \text{Uniform}(\mathcal{B})$$

We pick from a Poisson distribution because we need to sample independent random events occurring over a period of time with a fixed probability of happening at any time.

The durations for breakdowns are picked from lognormal distributions following the research conducted by Kline [4], who analysed various empirical electronic and mechanical disruption datasets and verified that the lognormal distribution is a suitable descriptor for corrective maintenance repair time.

The binary breakdown durations are then modelled as:

$$D_{\text{binary}} \sim \text{Lognormal}\left(D_{\text{binary},50}, D_{\text{binary},90}\right).$$

where we convert the 50th and 90th percentile parameters to the underlying normal mean and standard deviation.

The 50th percentile of a lognormal distribution is its median, so

$$\exp(\mu) = D_{\text{binary},50}.$$

The 90th percentile is given by

$$D_{\text{binary},90} = \exp\left(\mu + z_{0.90}\sigma\right),$$

where $z_{0.90} \approx 1.2816$.

Solving for μ and σ gives

$$\mu = \ln\left(D_{\text{binary},50}\right)$$

and

$$\sigma = \frac{\ln\left(D_{\text{binary},90}\right) - \ln\left(D_{\text{binary},50}\right)}{z_{0.90}}.$$

For partial durations, we first sample the severity of the breakdown with the formula:

$$\rho = \rho_{\min} + (1 - \rho_{\min}) * B,$$

where

$$B \sim \text{Beta}(\alpha, \beta)$$

is a beta distribution. The beta distribution was chosen because the outputs of the beta distribution are already in the range $[0, 1]$ and tuning the parameters gives significant freedom over the shape of the distribution. Here ρ means the remaining productivity of the berth. For binary breakdowns, the severity is by default $\rho = 0$. On the other hand, $\rho = 1$ means that the berth is fully open.

Partial breakdown duration is then conditionally sampled as:

$$D_{\text{partial}} \sim \text{Lognormal}\left(p, \gamma, D_{\text{partial},50}, D_{\text{partial},90}\right).$$

The median is adjusted according to the sampled productivity:

$$\mu(\rho) = \ln\left(D_{\text{partial},50}\right) - \gamma \ln(\rho),$$

where the γ of the distribution is used as a correlation coefficient of the severity and duration. The standard deviation is calculated the same way as for the binary breakdowns.

Many partial or binary breakdowns can overlap for the same berth at the same time. Let $\mathcal{E}_b(t)$ denote the set of active breakdown events affecting berth b at time t:

$$\mathcal{E}_b(t) = \left\{e \ : \ b_e = b, \ s_e \leq t < s_e + D_e\right\}.$$

The effective severity of berth b at time t is then defined as the product of the productivity factors of all active breakdowns:

$$\rho_b(t) = \prod_{e \in \mathcal{E}_b(t)} \rho.$$

ρ_b is the effective severity of the berth at any time. In particular, the formula for the new handling times during the timeslice is:

$$h_{broken} = h_{operational} / \rho_b$$

If $\rho_b = 0$ the berth is in a fully broken down state during the timeslice.

4 Experimental Setup

4.1 Evaluation

The model is evaluated with respect to four heuristics:

- **FCFS** assigns the earliest-arrived vessel to the available berth with the shortest processing time.
- **SPT** always selects the vessel-berth pair with the smallest handling time.
- **Priority** dispatches vessels in decreasing priority order, assigning each to the available berth with the shortest processing time.
- **WTSP** ranks all arrived-vessel/free-berth pairs by p/h and selects the highest-scoring pair.

WTSP is used as a benchmark across all experiment runs and results are calculated as a ratio of the service costs of the current run with respect to the benchmark costs in WTSP.

4.2 Experimental Process

The experiment evaluation loop can be found in Figure 1. Firstly, the trained agent is supplied with a starting state graph containing the instance-specific vessel and berth counts, vessel arrival times and all of the pre-computed state graph parameters. Events are encoded as upcoming vessel arrival or berth finish times in ascending order.

We then generate the berth breakdown schedule using the methodology described in Subsection 3.4. Each event is a three-dimensional vector containing the starting time, duration and severity: (s_i, D_i, ρ_i). The breakdown schedule is identical for the different heuristics / agent in one instance but ten different seeds are used to calculate the sample mean and standard deviation. The events are injected into the event queue of the agent, but none of the pre-computed parameters are updated to let the model know of the existence of the breakdowns.

This injection of events ensures the **Reactivity** and **Disruption response flexibility** assumptions discussed earlier in a natural way, using the existing agent. The assumption of **Occupancy-independent breakdowns** is ensured when generating the breakdown schedule (as the scheduler is unaware of occupied berths and vessels). In addition, after firing the berth breakdowns, the **Breakdowns do not erase progress** assumption is ensured by manually recalculating all of the berth handling times for a vessel.

After each event, an action log is appended with relevant information such as step count, current time, vessel priority, action and reward delta. Gantt diagrams (an example of which you can see in Figure 2) are optionally generated for instances where interesting or novel results require further analysis.

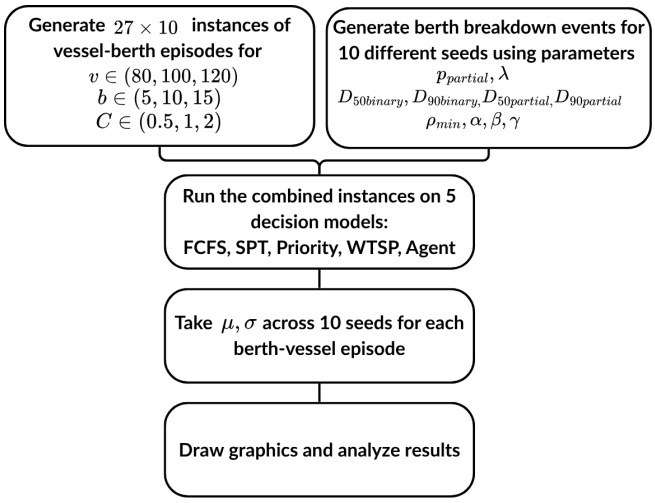


Figure 1: Experiment evaluation process diagram

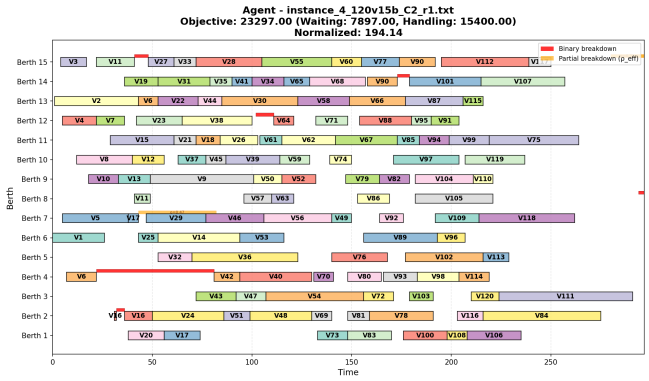


Figure 2: 5 berth 80 vessel example instance Gantt diagram with $\lambda = 0.02, p_{partial} = 0.2$

5 Results

This section presents the results of the experiment runs based on the setup described in Section 4. The full action log and instance generation results are available in the project's GitHub repository [5].

5.1 Breakdown frequency analysis

The first experiment analyses the effect of breakdown frequency. In Figure 3 we can see the performance of the binary berth breakdowns when varying the breakdown frequency. The results show that only the agent outperforms the heuristics, consistently staying below the baseline. The other heuristics perform considerably worse. Only the SPT heuristic is somewhat competitive, with a ~6-13% increase. Compared to WTSP, the Priority and FCFS heuristics perform considerably worse, exhibiting performance losses of several hundred percent. It is also noteworthy that Priority demonstrates better resilience under heavy breakdowns, whereas FCFS experiences a performance degradation peak at $\lambda \approx 0.12$.

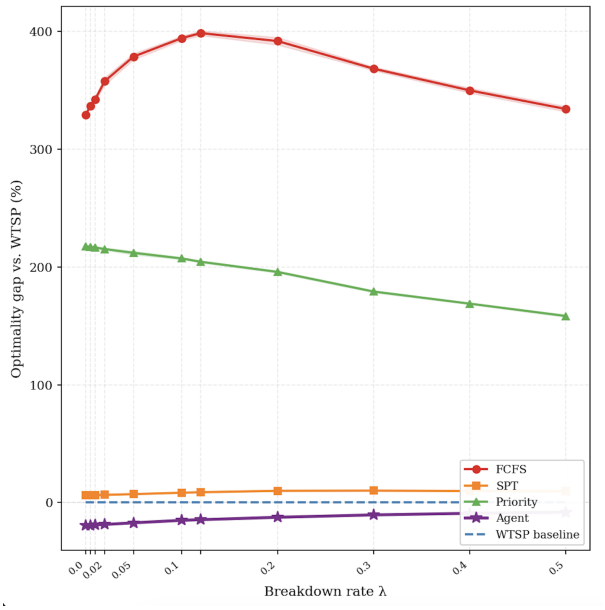


Figure 3: Heuristic performance versus agent under binary breakdown frequency change

Let us zoom in more closely and track the agent's performance in comparison to the baseline in Figure 4. We will also add the partial breakdowns benchmark with $\rho \sim \text{Beta}(2, 5)$ rescaled to $[0.1, 1]$, $\gamma = 1$ (severe breakdown skewed distribution), duration D_{50}=10, D_{90}=30. The results show that the agent's performance slightly degrades when increasing the breakdown rate, but even in the worst case it maintains a 6.4% advantage.

5.2 Breakdown duration analysis

The breakdown duration performance of the model can be seen in Figure 5. We can see that neither the agent nor the heuristics have a substantial change in performance for varying duration changes. The partial duration parameters chosen were the same as for the previous experiment.

5.3 Breakdown analysis by different priority classes

Let us now analyse how the agent prioritises different priority classes of vessels when faced with binary breakdowns. In Figure 6 with $\lambda = 0.02$ we can see that both the agent and the benchmark relatively increase the waiting time for the higher priority classes, but for the agent, the increase is even bigger than WTSP. This can explain why the performance of the model degrades relative to the benchmark.

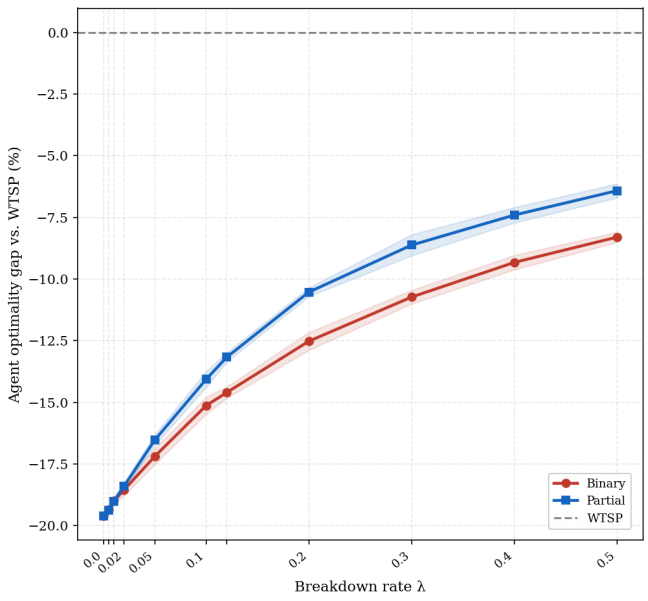
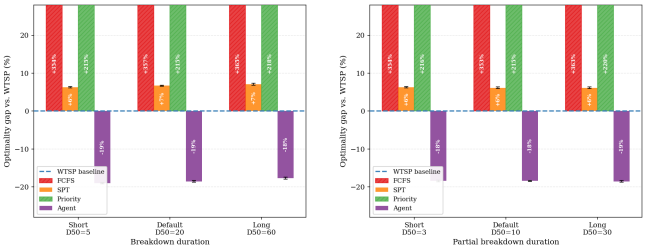


Figure 4: Agent performance under binary and partial breakdown frequency change



(a) Agent performance under binary breakdowns duration change

(b) Agent performance under partial breakdowns duration change

Figure 5: Agent performance under different breakdown durations.

5.4 Joint severity and frequency analysis

Figure 7 shows the agent's performance under different severity and frequency combinations. The experiment was done under full partial breakdowns with the parameters $\rho \sim \text{Beta}(2, 5)$, duration D_{50}=10, D_{90}=30, $\gamma = 1$. The experiment shows that lowering the ρ_{min} degrades the model's performance but breakdown frequency is still the dominant factor.

5.5 Severity distribution shape analysis

Figure 8 analyses how the shape of the severity distribution impacts the agent's performance. The parameters used were $\lambda = 0.02$, $\rho \sim \text{Beta}(2, 5)$ rescaled to $[0.1, 1]$, duration D_{50}=10, D_{90}=30. The results indicate that practically there is no difference in what shape the severity distribution takes on: uniform, skewed to be more or less severe, U-shaped. The agent performs similarly in all of the trials.

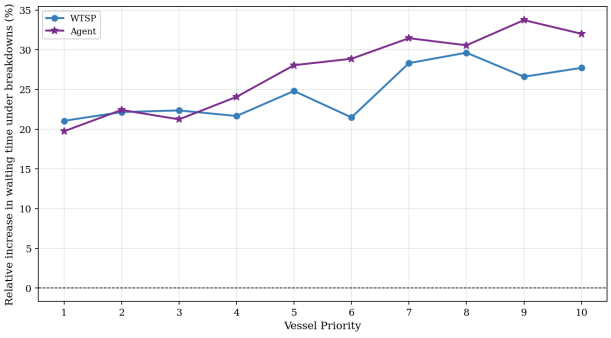


Figure 6: Waiting time increase by priority class under binary breakdowns

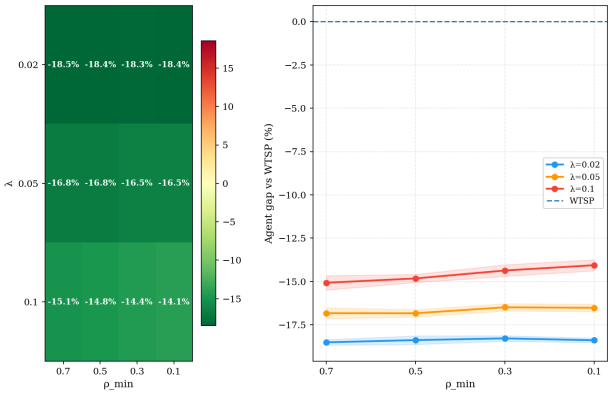


Figure 7: Agent performance versus baseline for different severity and frequency combinations

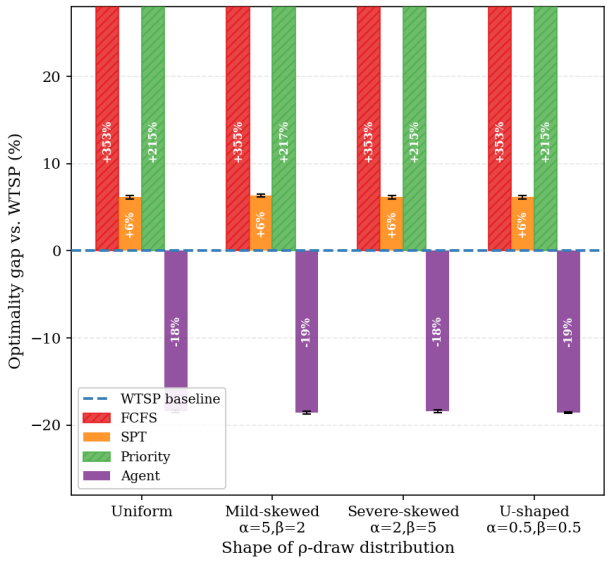


Figure 8: Performance of Agent and Heuristics versus baseline for different beta distribution shapes

5.6 Severity and duration coupling analysis

Figure 9 shows the effect of different duration–severity coupling parameters on model robustness. We test how different coupling parameters impact the robustness. The breakdown configuration was chosen as $\lambda = 0.02$, $\rho \sim \text{Beta}(2, 5)$ rescaled to $[0.1, 1]$, duration D_{50}=10, D_{90}=30. The results show that coupling does not have a major impact on the performance of the model.

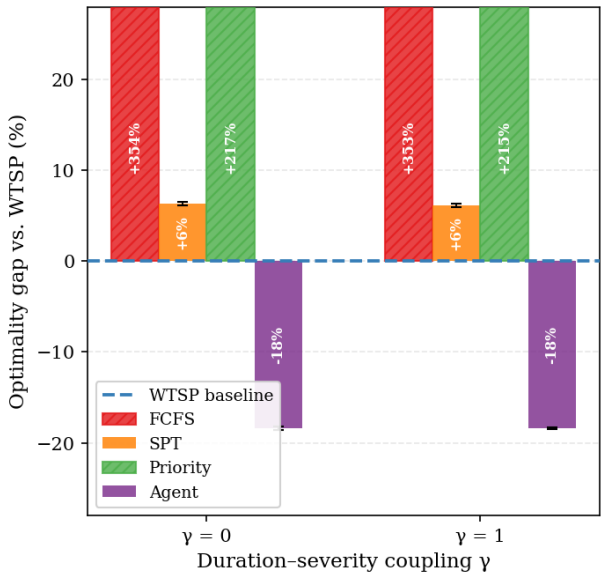


Figure 9: Agent performance versus baseline for different coupling levels

6 Responsible Research

This research was developed exclusively in a simulated environment and does not contain any personal information or commercially sensitive port data. The data was generated randomly and does not reflect any specific instance of real port operations. Therefore, the main responsible research risks come from modelling assumptions, reproducibility and data storage.

The breakdown model is an abstraction of real port operations. Although the generated breakdowns are designed to represent operational uncertainty, real disruptions may depend on factors such as weather, equipment condition, labour availability, safety constraints, and interactions between different parts of port infrastructure. As a result, the conclusions of this paper should be interpreted only within the assumptions of the simulation. The results show how the trained agent behaves under the defined breakdown model, but they should not be treated as direct evidence that the model is ready for deployment in real port operations.

To support reproducibility, the experiments use controlled random generation, fixed seeds, repeated runs, and identical breakdown schedules when comparing the agent against heuristic baselines. The generated instances, action logs, Gantt diagrams, code and experiment generation scripts are all stored in the project's GitHub repository [5]. This ensures that future experiments can be expanded and repeated.

7 Discussion

At the breakdown frequency of 0.02, the model retains 95% of its advantage relative to WTSP. Frequency of breakdowns dominates the performance of the model: changing frequency reduces the advantage from ∼19.6% to ∼6.4%, while changing duration decreases performance by at most 1%. No performance collapse can be observed in the agent across different experiments. Regarding breakdown splits, partial breakdowns perform slightly worse than binary breakdowns. A hypothesis for this behaviour could be that the agent can interpret binary breakdowns as ghost vessels in berths, whereas partial breakdowns only increase waiting times and do not block the berth entirely. Since partial breakdowns were not included during training, the agent may handle them less effectively. Under disruptions, it was observed that the agent disproportionately increases waiting times for high-priority vessels, suggesting that this inefficiency leads to the performance decrease.

8 Conclusions and Future Work

This paper set out to evaluate whether a trained GNN-based reinforcement learning agent for the DBAP retains its scheduling advantage when subjected to berth breakdowns. Across a comprehensive sweep of breakdown types, frequencies, durations and severities, we answered all of the research questions we set out to answer. The results confirm the agent generalises robustly beyond data it was trained on, though not without nuance.

Firstly, from the different experiments conducted, we can see that the model still heavily outperforms all evaluated heuristics, with FCFS, Priority and SPT all degrading and having higher service costs than WTSP by a wide margin. Only SPT stays competitive, incurring service costs 6–13% higher than the WTSP baseline. On the other hand, the agent consistently outperforms WTSP by ∼6.4-20% **(RQ)**.

Duration and severity do produce mild degradations in the agent's performance, but breakdown frequency is the driving factor **(SQ1)**.

The agent does not produce significant performance deterioration under berth breakdowns, but its performance slightly decreases relative to WTSP under high-frequency breakdowns **(SQ2)**. No threshold effect was observed for any configuration of parameters.

The decrease in performance could be attributed to the fact that the agent relatively increases waiting times for high-priority vessels **(SQ3)**.

In general, the agent demonstrates that it can handle various port disruption models. This shows that the agent is robust and demonstrates substantial strength even on data that it was not trained on.

Despite the promising results, several limitations remain, each pointing to directions for future research:

- The first limitation is that the agent was not compared against an exact solver. An exact method, such

as a mixed-integer programming model, could provide a clearer estimate of the optimality gap for small instances. However, due to the dynamic nature of the breakdowns, we would have to recompute the optimal solution each time they happened. This was not done due to computational and time constraints. A partial solution would be to provide the solver with a heuristic's or agent's warm start and given a timeout, let it find a better one.

- The second limitation is that real port data was not used. The vessel instances and breakdown events were generated synthetically, so they do not correspond to any specific port or historical disruption pattern. Nevertheless, the distributions were chosen to approximate real-world behaviour: breakdown berths were chosen uniformly, with frequency sampled from a Poisson process with exponential inter-arrival times, durations follow the lognormal distribution and severities are sampled from a flexible beta distribution. Further research would be needed to assess the model on empirical distributions.
- The third limitation is that the main reinforcement-learning model evaluated in this paper is based on the work of March Moya et al. [9], which, at the time of writing, is unpublished. As a result, the underlying model architecture, training procedure, and original benchmark results cannot yet be fully verified through a public peer-reviewed source. This paper should therefore be understood as an extension and stress-test of that model under berth breakdowns.

Generative AI disclosure

This paper was written with the help of generative AI tools. Specifically, Claude Sonnet 4.6 was used for language editing, clarification of technical writing, and text formatting in this paper. No AI was used for designing the experiments, architecture or analysis of the research. The author takes full responsibility for the design, implementation and conclusions of the work.

References

[1] K. Buhrkal, S. Zuglian, S. Ropke, J. Larsen, and R. Lusby. Models for the discrete berth allocation problem: A computational comparison. *Transportation Research Part E: Logistics and Transportation Review*, 47(4):461–473, 2011.

[2] A. Imai, E. Nishimura, and S. Papadimitriou. The dynamic berth allocation problem for a container port. *Transportation Research Part B: Methodological*, 35(4):401–417, 2001.

[3] S. Jo and I. Moon. A hierarchical reinforcement learning approach for real-time berth allocation and quay crane scheduling. *International Journal of Production Research*, 63(24):10027–10052, 2025.

[4] M. B. Kline. Suitability of the lognormal distribution for corrective maintenance repair times. *Reliability Engineering*, 9(2):65–80, 1984.

[5] T. Kuklys. Evaluating a trained dynamic berth allocation model under berth breakdowns. https://github.com/TauKuk/BRP-RL-BAP, 2026. GitHub repository.

[6] S. Li, P. Xu, Z. Ding, Z. Kong, and W. Luo. Evolutionary reinforcement learning with density-based behavioral diversity enhancement for berth allocation and crane assignment. *Swarm and Evolutionary Computation*, 100:102241, 2026.

[7] Y. Lv, M. Zou, J. Li, and J. Liu. Dynamic berth allocation under uncertainties based on deep reinforcement learning towards resilient ports. *Ocean & Coastal Management*, 252:107113, 2024.

[8] X. Lyu, R. R. Negenborn, X. Shi, and F. Schulte. A collaborative berth planning approach for disruption recovery. *IEEE Open Journal of Intelligent Transportation Systems*, 3:153–164, 2022.

[9] C. March Moya, F. Schulte, K. Tierney, and N. Yorke-Smith. A reinforcement learning approach for the dynamic berth allocation problem. Working paper; accepted and will be published at LOGMS 2026, 2026.

[10] L. P. Prencipe and M. Marinelli. A novel mathematical formulation for solving the dynamic and discrete berth allocation problem by using the bee colony optimisation algorithm. *Applied Intelligence*, 51(7):4127–4142, 2021.

[11] F. Rodrigues and A. Agra. Berth allocation and quay crane assignment/scheduling problem under uncertainty: A survey. *European Journal of Operational Research*, 303(2):501–524, 2022.

[12] T. Smith and H. Francis. Port congestion, waiting times and operational efficiency. Technical report, UCL Energy Institute and UMAS, December 2024. Available at: https://www.u-mas.co.uk/wp-content/uploads/2024/12/Port-Congestion-Report-v1.8.pdf.

[13] Transport & Environment. EU ports' climate performance. Briefing, Transport & Environment, February 2022. Available at: https://www.transportenvironment.org/uploads/files/2202_Port_Rankings_briefing-1.pdf.

[14] N. Umang, M. Bierlaire, and A. L. Erera. Real-time management of berth allocation with stochastic arrival and handling times. *Journal of Scheduling*, 20:67–83, 2017.

[15] H. Zheng, Z. Wang, and H. Liu. The integrated rescheduling problem of berth allocation and quay crane assignment with uncertainty. *Processes*, 11(2):522, 2023.