

A Heuristic Algorithm Based on Beam Search and Iterated Local Search for the Maritime Inventory Routing Problem

Rafael Meirelles

Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro
Rua Marquês de São Vicente, 225 - Gávea, Rio de Janeiro / RJ, 22.451-900
rafaelpmeirelles@aluno.puc-rio.br

Nathalie Sanghikian

Departamento de Engenharia Industrial, Pontifícia Universidade Católica do Rio de Janeiro
Rua Marquês de São Vicente, 225 - Gávea, Rio de Janeiro / RJ, 22.451-900
nat_sanghikian@aluno.puc-rio.br

Anand Subramanian

Departamento de Sistemas de Computação, Centro de Informática, Universidade Federal da Paraíba
Rua dos Escoteiros, s/n - Mangabeira, João Pessoa / PB, 58.058-600
anand@ci.ufpb.br

Rafael Martinelli

Departamento de Engenharia Industrial, Pontifícia Universidade Católica do Rio de Janeiro
Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro
Rua Marquês de São Vicente, 225 - Gávea, Rio de Janeiro / RJ, 22.451-900
martinelli@puc-rio.br

ABSTRACT

This work uses a pure heuristic approach to address the deterministic Maritime Inventory Routing Problem (MIRP). The method combines a Beam Search algorithm, guided by a greedy heuristic and refined by an Iterated Local Search (ILS) with local search procedures. A novel solution representation and auxiliary data structures are introduced to ensure feasibility and efficiency during the search. Computational experiments on MIRPLib's Group 2 instances show that the approach yields high-quality solutions within reasonable times, improving best-known results for ten out of 72 instances. The method is flexible, tuned to available runtime, and competitive with state-of-the-art matheuristics, especially for large-scale instances.

KEYWORDS. Maritime Inventory Routing. Heuristics. Beam Search. Iterated Local Search.

MH - Metaheuristics, EN&OG – PO in the Energy, Oil and Gas Area

1. Introduction

The maritime transport sector handles over 80% of global trade by volume [United Nations Conference on Trade and Development (UNCTAD), 2023], making logistical improvements crucial for cost and efficiency. The Maritime Inventory Routing Problem (MIRP) integrates inventory management and maritime routing, considering constraints such as ship capacities, berth availability, and port inventory bounds. Its planning horizon is typically long, and the vessels involved often have heterogeneous characteristics, making the problem more like a scheduling problem than traditional vehicle routing.

Although exact and hybrid methods solve some MIRP instances, they often require specific formulations and are computationally expensive. Many advances rely on metaheuristics that combine local heuristics with mathematical programming solvers [Papageorgiou et al., 2018; Friske et al., 2022; Friske and Buriol, 2020], limiting their use in fast-paced operations [INFORMS, 2024]. This work proposes a pure metaheuristic for the deterministic MIRP: a Beam Search (BS) guided by a greedy heuristic, followed by local search (LS) and Iterated Local Search (ILS). A novel solution representation avoids common ambiguities, allowing for practical solution space exploration. The BS constructs feasible solutions while balancing breadth and depth, and LS + ILS refine them. Tested on MIRPLib's Group 2 instances [Papageorgiou et al., 2014], our approach improved ten out of the 72 best-known results, demonstrating its viability as a scalable, solver-independent alternative for large-scale maritime planning scenarios.

2. Problem Description

The MIRP determines the routes and schedules of vessels to transport a single product between production and consumption ports over a fixed horizon, minimizing total inventory holding and sailing costs while meeting operational constraints. We consider the deterministic, single-product core model from MIRPLib [Papageorgiou et al., 2014], including inventory limits, loading / unloading windows, berth availability, and vessel-specific attributes (capacity, speed, fuel consumption). Horizons typically span 120-360 days, requiring synchronization between vessel schedules and port operations. This paper focuses on MIRPLib's Group 2 instances where vessels cannot sail partially loaded, a practical constraint in oil and gas logistics. Ports have deterministic production and consumption rates, and inventory violations incur penalties. A feasible solution specifies the sequence of calls for the vessels and ports, the times, and the quantities loaded / unloaded, while respecting inventory, berth, capacity, and draft limits. The routing, timing, and inventory interplay makes MIRP highly constrained and challenging to solve using standard techniques.

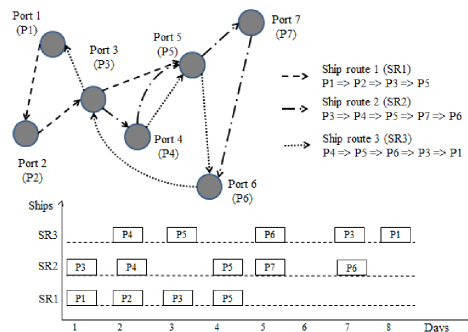


Figure 1: Example of a MIRP solution. Adapted from De et al. [2017].

3. Proposed Methodology

The algorithm uses a three-phase structure: constructive Beam Search, local search, and Iterated Local Search (ILS), as shown in Figure 2. It is purely heuristic and does not rely on mathematical programming solvers.

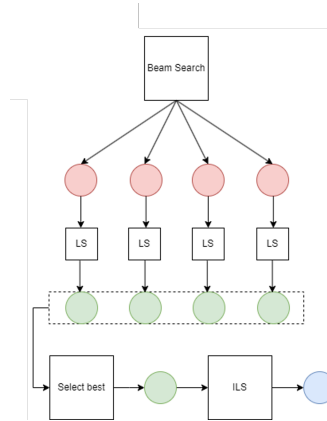


Figure 2: The metaheuristic algorithm. Source: The Author

3.1. Solution Representation

A key MIRP challenge is an effective solution representation. Traditional per-vessel route lists (list-of-lists) are ambiguous for MIRP berth timing, as schedules depend on absolute timing, not just route order. We use a single, chronologically ordered list of calls (*vessel*, *port*), eliminating ambiguities and allowing for consistent constraint evaluation over time. Although this introduces redundancy (multiple permutations encoding the same plan), we mitigate this issue by maintaining auxiliary information during local search. For each call, we store the positions of previous and subsequent visits by the same vessel and port. This information allows to restrict moves to structurally distinct and potentially improving changes. They are only constructed and maintained during the local search phase when needed to reduce overhead.

3.2. Beam Search and Greedy Completion

Beam Search (BS) maintains N (beam width) active nodes per depth. Starting with an empty solution, it iteratively expands each node by generating valid calls that can be appended. Let a node be a sequence of k calls. At depth $k + 1$, successors are formed by appending every feasible call (any vessel / port respecting current capacity, inventory, and berth constraints). The number of succeeding nodes are limited by the parameter w [Morais et al., 2024]. Each partial solution is evaluated by a greedy completion heuristic that simulates the remaining horizon, inserting calls to prevent inventory violations by prioritizing the port with the earliest violation and the vessel serving it first. This biased search towards inventory feasibility is crucial for Group 2 instances. Controlled stochasticity is used to promote diversity: ports are sampled via a softmax distribution over violation times (higher probability for imminent violation), and vessel selection is randomized based on earliest arrival using a biased rule. Each partial solution is completed q times (once deterministically, $q - 1$ times stochastically). The median cost ranks the candidates, balancing reliability and exploration. In experiments, using stochasticity only in port selection (with the vessel choice being deterministic) was most stable. The parameters N, w, q are calibrated to balance performance and efficiency.

3.3. Local Search and Iterated Local Search

The best BS solution undergoes a Variable Neighborhood Descent (VND), exploring six predefined MIRP-specific neighborhoods: *Swap* (exchanges two calls); *Relocate* (moves a call); *Replace* (substitutes a port with another of the same type, balancing producer-consumer aspects); *Insert* (adds a new call at the end); *Delete* (removes a call and the next by the same vessel for feasibility); and *SwapPort* (exchanges destinations of two vessels, preserving port type). Moves maintain inventory / berth feasibility and avoid redundant changes. Auxiliary information filter unproductive

/ symmetric modifications. Neighborhoods are randomly applied with the first improvement strategy, restarting the sequence, balancing intensification, and diversification. ILS perturbs the VND local optimum (using the same operators) to escape local minima and re-applies local search. Acceptance uses a Simulated Annealing criterion ($e^{-\Delta/T}$), with a decaying temperature T , allowing escape from deep minima while gradually tightening acceptance. ILS parameters were calibrated using `Hypertuning.jl`. ILS proved to be essential for further quality improvement, especially in larger instances.

4. Computational Results

4.1. Setup

The experiments were performed on an Intel Core i7-12700H (2.7GHz, 32GB RAM, Ubuntu 22.04), single-threaded, and Julia 1.9. The runtime was limited per instance. The tests used all 72 Group 2 MIRPLib instances [Papageorgiou et al., 2014] (120, 180, or 360-day horizons), which are deterministic, single-product, no-split-loading MIRPs. Instances grouped by difficulty (Easy, Medium, Hard) per Papageorgiou et al. [2018], based on the average optimality gaps of state-of-the-art matheuristics. The gaps are relative to MIRPLib BKS (May 2024).

4.2. Parameter Optimization

Effective parameter calibration is critical in heuristic algorithms. The tuning process focused on Beam Search branching, greedy completions per node, and ILS configuration. The maximum number of children per node was tested for the Beam Search branching factor (w). Using $N = 100$ and 10 seeds on “Easy” instances, $w = 2$ yielded the best average (-0.70%) and median (-0.13%) GAPs. For Greedy Completions (q), $q = 3$ yielded the best average performance when the CPU time was kept approximately constant for evaluation (setting $N \propto 1/q$). For ILS Parameters, `HyperTuning.jl` was used (15 representative instances, five seeds, 72h total optimization). The best values found were: initial SA probability $sa_{in} = 0.79$, final SA probability $sa_{fi} = 0.01$, ILS iterations $\eta = 640$, non-improving iterations before restart $re = 4$, and perturbations per ILS round $pb = 2$.

4.3. Performance

The method was evaluated with varying $N \in \{10, 100, 1000, 10000\}$ and a 24-hour CPU limit per instance, using 640 ILS iterations (calibrated as above). The goal was to examine beam width, solution quality and time trade-offs, and analyze LS/ILS contributions.

Impact of Heuristic Components. For small N (e.g. $N = 10$), both LS and ILS substantially improve the initial BS output, with ILS reducing average GAPs by 2-4 percentage points across the Easy, Medium, and Hard classes, although at a significant increase in execution time. As N increases (e.g., to $N = 1000$), BS combined with LS already provides high-quality solutions, and ILS offers smaller marginal gains while increasing runtime. These results highlight the constructive phase’s strength with a wider beam. For large N , it alone provides high-quality solutions, thus reducing post-processing benefits.

Scaling Behavior and Detailed Results Increasing N consistently improves the quality of the BS solution, which carries over to the final results, as seen across the planning horizons in Tables 1, 2, and 3. Local Search (LS) provides consistent improvements across all N values, while the marginal contribution of ILS decreases for higher N , particularly when $N \geq 1000$. Result variability, indicated by interquartile ranges in objective values (not explicitly shown as box plots here due to space, but observed in the analysis), also tends to decrease as N grows. These tables present detailed GAPs for instances with 120, 180, and 360 periods across different N values and heuristic stages (BS, LS, ILS). They highlight instances where new best-known solutions (negative GAPs) were achieved, particularly with $N = 10000$. Ten new BKS were found overall. The “X%” entries denote configurations that have not been completed within the time limit or not run for comparison. Blank cells

typically indicate that the configuration with a smaller N already hit the time limit for that instance class or that the specific N value was not tested for that group due to excessive runtime.

Table 1: Results 120 periods

Instance	Class.	N = 10			N = 1000			N = 10000		
		BS Gap	LS Gap	ILS Gap	BS Gap	LS Gap	ILS Gap	BS Gap	LS Gap	ILS Gap
LR1_DR02_VC01_V6a	E	1.39%	0.00%	0.00%	1.39%	0.00%	0.00%	1.39%	0.00%	0.00%
LR1_DR02_VC02_V6a	E	3.93%	2.48%	2.46%	1.21%	0.01%	0.01%	0.82%	0.00%	0.00%
LR1_DR02_VC03_V7a	E	14.62%	11.87%	10.43%	1.90%	-0.09%	-0.10%	1.33%	-0.24%	-0.24%
LR1_DR02_VC03_V8a	E	16.11%	11.52%	9.22%	2.89%	0.48%	0.48%	1.69%	0.00%	0.00%
LR1_DR02_VC04_V8a	E	7.33%	2.26%	1.38%	3.02%	0.59%	0.46%	2.83%	0.15%	0.15%
LR1_DR02_VC05_V8a	E	3.54%	1.47%	0.94%	2.43%	-0.10%	-0.12%	2.25%	-0.30%	-0.30%
LR1_DR03_VC03_V10b	M	24.20%	19.56%	17.97%	5.68%	3.15%	2.85%	3.28%	0.21%	0.15%
LR1_DR03_VC03_V13b	E	10.58%	6.86%	3.71%	4.74%	2.18%	1.79%	3.05%	0.59%	0.50%
LR1_DR03_VC03_V16a	E	34.13%	30.74%	22.42%	12.69%	10.51%	9.30%	—	—	—
LR1_DR04_VC03_V15a	E	13.20%	5.75%	2.93%	5.57%	2.15%	1.88%	—	—	—
LR1_DR04_VC03_V15b	E	15.93%	10.07%	6.91%	2.77%	-0.83%	-1.17%	—	—	—
LR1_DR04_VC05_V17a	E	6.96%	3.22%	1.90%	4.96%	1.68%	1.34%	—	—	—
LR1_DR04_VC05_V17b	E	15.70%	11.18%	7.28%	2.67%	-1.26%	-2.16%	—	—	—
LR1_DR05_VC05_V25a	M	5.69%	2.37%	0.76%	3.35%	0.79%	0.63%	—	—	—
LR1_DR05_VC05_V25b	E	30.07%	25.82%	15.24%	7.46%	4.00%	2.42%	—	—	—
LR1_DR08_VC05_V38a	M	15.53%	8.33%	4.58%	5.48%	2.71%	2.06%	—	—	—
LR1_DR08_VC05_V40a	E	14.17%	7.09%	3.61%	7.32%	4.15%	3.48%	—	—	—
LR1_DR08_VC05_V40b	M	31.55%	22.40%	18.24%	—	—	—	—	—	—
LR1_DR08_VC10_V40a	M	12.78%	4.23%	2.18%	—	—	—	—	—	—
LR1_DR08_VC10_V40b	M	33.15%	22.75%	16.70%	—	—	—	—	—	—
LR1_DR12_VC05_V70a	H	24.06%	11.90%	6.95%	—	—	—	—	—	—
LR1_DR12_VC05_V70b	M	26.65%	11.28%	8.53%	—	—	—	—	—	—
LR1_DR12_VC10_V70a	H	16.94%	4.26%	2.59%	—	—	—	—	—	—
LR1_DR12_VC10_V70b	H	26.32%	6.48%	2.93%	—	—	—	—	—	—

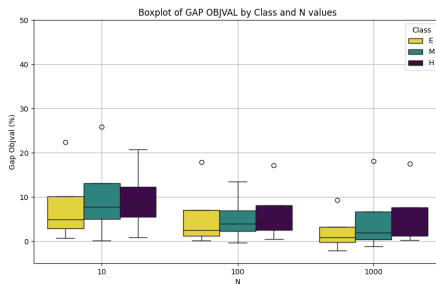
Table 2: Results 180 periods

Instance	Class.	N = 10			N = 1000			N = 10000		
		BS Gap	LS Gap	ILS Gap	BS Gap	LS Gap	ILS Gap	BS Gap	LS Gap	ILS Gap
LR1_DR02_VC01_V6a	E	0.90%	0.00%	0.00%	0.90%	0.00%	0.00%	0.90%	0.00%	0.00%
LR1_DR02_VC02_V6a	E	5.07%	3.95%	3.42%	X%	X%	X%	X%	X%	X%
LR1_DR02_VC03_V7a	M	14.72%	11.95%	11.55%	X%	X%	X%	X%	X%	X%
LR1_DR02_VC03_V8a	E	18.88%	15.18%	14.41%	X%	X%	X%	X%	X%	X%
LR1_DR02_VC04_V8a	M	5.61%	3.27%	2.81%	X%	X%	X%	X%	X%	X%
LR1_DR02_VC05_V8a	M	1.58%	0.30%	0.07%	X%	X%	X%	X%	X%	X%
LR1_DR03_VC03_V10b	M	14.18%	10.35%	8.39%	X%	X%	X%	X%	X%	X%
LR1_DR03_VC03_V13b	E	6.78%	2.72%	0.37%	X%	X%	X%	X%	X%	X%
LR1_DR03_VC03_V16a	M	34.05%	32.48%	25.77%	X%	X%	X%	—	—	—
LR1_DR04_VC03_V15a	M	13.80%	8.70%	5.76%	X%	X%	X%	—	—	—
LR1_DR04_VC03_V15b	E	9.32%	4.11%	2.01%	X%	X%	X%	—	—	—
LR1_DR04_VC05_V17a	M	5.14%	2.38%	1.49%	X%	X%	X%	—	—	—
LR1_DR04_VC05_V17b	M	12.36%	6.36%	4.83%	X%	X%	X%	—	—	—
LR1_DR05_VC05_V25a	M	5.09%	2.13%	0.61%	X%	X%	X%	—	—	—
LR1_DR05_VC05_V25b	M	33.85%	26.22%	19.99%	X%	X%	X%	—	—	—
LR1_DR08_VC05_V38a	H	26.87%	15.06%	11.19%	—	—	—	—	—	—
LR1_DR08_VC05_V40a	H	15.52%	9.71%	5.71%	—	—	—	—	—	—
LR1_DR08_VC05_V40b	M	25.52%	14.28%	10.56%	—	—	—	—	—	—
LR1_DR08_VC10_V40a	H	17.01%	10.91%	5.50%	—	—	—	—	—	—
LR1_DR08_VC10_V40b	H	27.32%	15.55%	11.24%	—	—	—	—	—	—
LR1_DR12_VC05_V70a	H	20.85%	9.15%	6.30%	—	—	—	—	—	—
LR1_DR12_VC05_V70b	H	28.28%	14.03%	10.99%	—	—	—	—	—	—
LR1_DR12_VC10_V70a	H	14.67%	4.92%	3.31%	—	—	—	—	—	—
LR1_DR12_VC10_V70b	H	29.55%	6.96%	2.00%	—	—	—	—	—	—

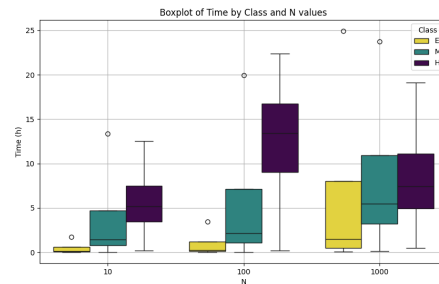
Table 3: Results 360 periods

Instance	Class.	N = 10			N = 1000			N = 10000		
		BS Gap	LS Gap	ILS Gap	BS Gap	LS Gap	ILS Gap	BS Gap	LS Gap	ILS Gap
LR1_DR02_VC01_V6a	E	17.60%	8.20%	6.31%	X%	X%	X%	X%	X%	X%
LR1_DR02_VC02_V6a	E	6.38%	5.44%	5.08%	X%	X%	X%	X%	X%	X%
LR1_DR02_VC03_V7a	H	19.82%	15.90%	14.51%	X%	X%	X%	X%	X%	X%
LR1_DR02_VC03_V8a	M	20.53%	17.46%	15.57%	X%	X%	X%	X%	X%	X%
LR1_DR02_VC04_V8a	M	7.36%	4.34%	3.63%	X%	X%	X%	X%	X%	X%
LR1_DR02_VC05_V8a	H	2.84%	1.54%	0.86%	X%	X%	X%	X%	X%	X%
LR1_DR03_VC03_V10b	H	21.12%	18.35%	16.18%	X%	X%	X%	X%	X%	X%
LR1_DR03_VC03_V13b	M	10.11%	5.29%	2.96%	X%	X%	X%	X%	X%	X%
LR1_DR03_VC03_V16a	M	29.87%	27.86%	25.81%	X%	X%	X%	—	—	—
LR1_DR04_VC03_V15a	M	17.41%	10.38%	4.35%	X%	X%	X%	—	—	—
LR1_DR04_VC03_V15b	M	17.58%	10.12%	4.95%	X%	X%	X%	—	—	—
LR1_DR04_VC05_V17a	H	5.43%	2.32%	1.45%	X%	X%	X%	—	—	—
LR1_DR04_VC05_V17b	M	20.53%	15.05%	11.74%	X%	X%	X%	—	—	—
LR1_DR05_VC05_V25a	H	8.36%	5.05%	1.89%	X%	X%	X%	—	—	—
LR1_DR05_VC05_V25b	H	30.75%	28.46%	20.71%	—	—	—	—	—	—

Figure 3 illustrates the distribution of GAPObjval (Panel a) and runtime in hours (Panel b) for different N values across the instance classes (Easy, Medium, Hard). As expected, Easy instances generally require significantly less computational time and yield better objective values across all settings. Medium instances tend to show a higher variance in solution quality, especially when using smaller values of N . Hard instances are consistently the most computationally demanding and exhibit less sensitivity to increases in N regarding GAP reduction, although their CPU time requirements increase sharply with larger N . These results highlight the trade-off between solution quality and computational effort, particularly for the more challenging problems.



(a) GAPObjval distribution.



(b) Runtime (hours) distribution.

Figure 3: Distribution of (a) GAPObjval and (b) Runtime (hours) by instance class and N values.

To further evaluate time-sensitivity, Figure 4 shows the evolution of GAPObjval over time for different N values across Easy, Medium, Hard, and all instance classes, comparing our approach against the results of Munguía et al. [2019] (data points for their work are illustrative based on their reported performance at 500 and 3600 seconds). Although their method often reports better raw GAPs within these short timeframes, it is crucial to contextualize these results. Their experiments were conducted on a 96-core Xeon X5650 cluster, whereas ours ran on a single thread of a significantly newer i7-12700H. Considering that each core of the i7-12700H delivers substantially more performance than an older X5650 core (industry benchmarks suggest a factor of 2.5x or more in single-thread performance, leading to an estimated 38x more total processing power for their setup), our method demonstrates highly competitive solution quality progression when normalized for computational budget. Our approach shows a clear trend of improvement with increasing N ,

and given more time or parallel execution (which BS and ILS are capable of), further gains would be expected.

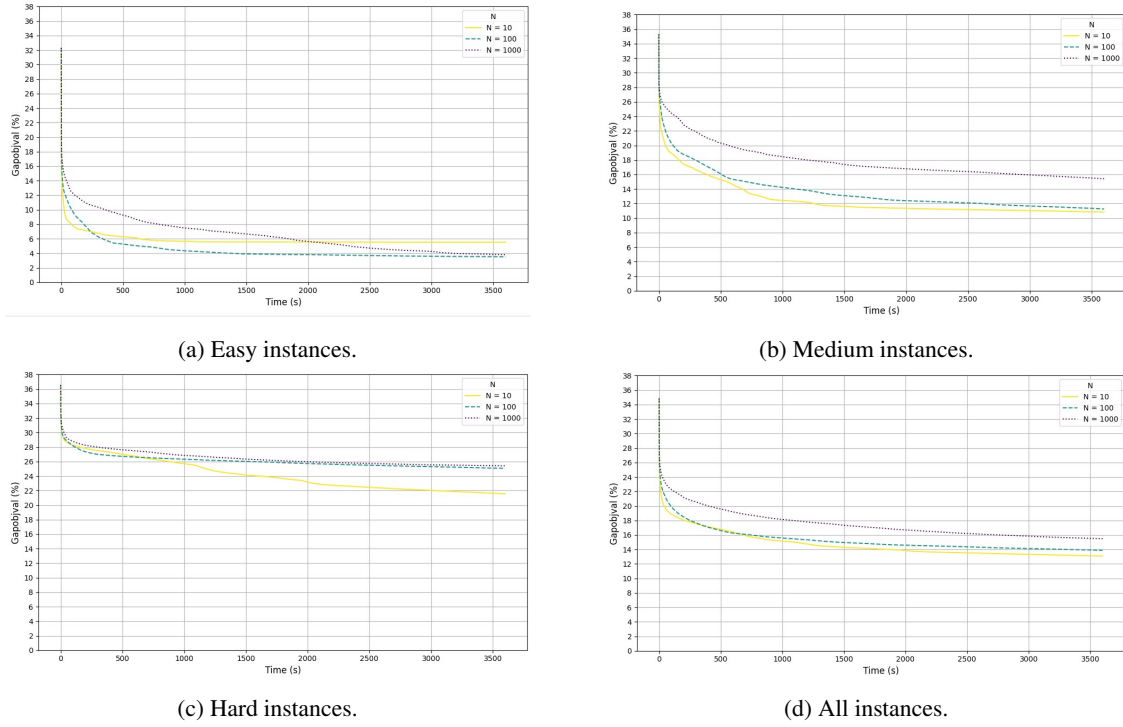


Figure 4: GAPObjval over time by N values and instance classes. Illustrative comparison points for Munguía et al. [2019] (500s, 3600s) shown if available.

The analysis confirms that increasing N leads to better solution quality and lower variance at the cost of longer runtimes. ILS proves essential for small N , but its contribution decreases as the quality of the beam search improves. Comparisons with state-of-the-art methods highlight the relevance of computational constraints, underscoring the competitiveness of our approach even under tighter resource settings.

5. Conclusion and Future Work

This paper introduced a pure metaheuristic for the deterministic Maritime Inventory Routing Problem (MIRP) that avoids mathematical programming solvers. The method combines Beam Search, Local Search, and Iterated Local Search, employing a novel solution representation to eliminate berth timing ambiguities and support efficient neighborhood exploration. Tested on MIRPLib's Group 2 instances, the approach proved competitive and flexible, improving ten best-known solutions. Its performance is adjusted to the available runtime, and despite the known difficulty of obtaining high-quality solutions quickly for this instance group [Papageorgiou, 2013], our observations show that calibrated parameters (e.g., $N = 1000$, ILS with 640 iterations) yield high-quality results even under limited computational budgets.

The experiments revealed that the contributions of the heuristic components vary with N . For low N values, LS and ILS provide significant improvements. As N increases, the Beam Search becomes more powerful, diminishing the marginal impact of ILS. At $N = 10000$, the constructive phase and LS account for most of the improvements.

Future work will focus on improving the evaluation strategies for partial MIRP solutions and reducing the high computational cost of greedy completion, potentially through parallelization.

Investigating adaptive tuning N and new ILS neighborhoods or memory mechanisms for enhanced diversification also holds promise.

References

- De, A., Kumar, S. K., Gunasekaran, A., and Tiwari, M. K. (2017). Sustainable maritime inventory routing problem with time window constraints. *Engineering Applications of Artificial Intelligence*, 61:77–95. URL <https://api.semanticscholar.org/CorpusID:10369982>.
- Friske, M. W. and Buriol, L. S. (2020). A multi-start algorithm and a large neighborhood search for a maritime inventory routing problem. In *2020 IEEE Congress on Evolutionary Computation (CEC)*, p. 1–8. IEEE.
- Friske, M. W., Buriol, L. S., and Camponogara, E. (2022). A relax-and-fix and fix-and-optimize algorithm for a Maritime Inventory Routing Problem. *Computers & Operations Research*, 137:105520.
- INFORMS (2024). Metaheuristics in Optimization: Algorithmic Perspective. OR/MS Tomorrow. URL <https://www.informs.org/Publications/OR-MS-Tomorrow/Metaheuristics-in-Optimization-Algorithmic-Perspective>. Accessed: 2024-05-28.
- Morais, R., Bulhões, T., and Subramanian, A. (2024). Exact and heuristic algorithms for minimizing the makespan on a single machine scheduling problem with sequence-dependent setup times and release dates. *European Journal of Operational Research*, 315(2):442–453.
- Munguía, L. M., Ahmed, S., Bader, D. A., Nemhauser, G. L., Shao, Y., and Papageorgiou, D. J. (2019). Tailoring parallel alternating criteria search for domain specific MIPs: Application to maritime inventory routing. *Computers & Operations Research*, 111:21–34.
- Papageorgiou, D. J. (2013). A Mixed-Integer Linear Programming Formulation for "Group 2" Maritime Inventory Routing Instances. Technical report, School of Industrial and Systems Engineering, Georgia Institute of Technology.
- Papageorgiou, D. J., Cheon, M.-S., Harwood, S., Trespalacios, F., and Nemhauser, G. L. (2018). Recent progress using matheuristics for strategic maritime inventory routing. In Ronay, M. E., editor, *Modeling, Computing and Data Handling Methodologies for Maritime Transportation*, volume 8 of *Springer Series in Supply Chain Management*, p. 59–94. Springer.
- Papageorgiou, D. J., Nemhauser, G. L., Sokol, J., Cheon, M.-S., and Keha, A. B. (2014). MIRPLib – A library of maritime inventory routing problem instances: Survey, core model, and benchmark results. *European Journal of Operational Research*, 235(2):350–366.
- United Nations Conference on Trade and Development (UNCTAD) (2023). Review of Maritime Transport 2023. Technical report, United Nations, New York. URL <https://unctad.org/publication/review-maritime-transport-2023>. Accessed: 2024-05-28.