Enterprise Development & Microfinance, 36:3, 219–234

A Reinforcement-Learning-Tuned Hybrid GA–PSO Algorithm for Continuous-Time Crude Oil Scheduling with Quality and Logistics Constraints

BOUKELKOUL ZAKARIA, MOUSS KINZA NADIA

Abstract: Crude-oil scheduling combines discrete vessel-to-tank assignments with continuous timing and nonconvex blend-quality relations. This study presents an event-driven assignment-and-timing formulation with inventory holding, demurrage, sulfur and capacity penalties, CDU starvation, and melt-class-switch flushing costs. A hybrid algorithm embeds short particle swarm optimization (PSO) searches within a genetic algorithm (GA), while tabular Q-learning adapts mutation and inertia using stagnation and diversity. The reported experiments compare the hybrid, standalone GA, standalone PSO, and a greedy heuristic on 20 paired synthetic instances at each of three scales: 6, 12, and 20 vessels. The hybrid has the lowest reported mean cost at every scale and the highest or joint-highest feasibility rate. All reported one-sided Wilcoxon comparisons have $p < 0.001$; Friedman tests also indicate rank differences. Feasibility nevertheless falls to 75% for the hybrid at the largest scale. A 3×3 factorial study finds no significant main or interaction effects for the two tested search-effort parameters, without establishing equivalence. These results concern synthetic benchmarks rather than refinery operating data. Because embedded PSO adds evaluations, equal-generation comparisons do not isolate the benefits of hybridization or reinforcement learning. The formulation uses instantaneous parcel additions, and validation of finite-rate unloading, operational feasibility, and industrial cost parameters remains necessary before deployment.

Keywords: Crude Oil Scheduling; Hybrid Metaheuristics; Genetic Algorithm; Particle Swarm Optimization; Reinforcement Learning; Continuous-Time MINLP.

Boukelkoul Zakaria: z.boukelkoul@univ-batna2.dz; boukelkoulzakaria@gmail.com

Mouss Kinza Nadia: k.mouss@univ-batna2.dz

Laboratory of Automation and Manufacturing, Industrial Engineering Department, Faculty of Technology, University of Batna 2 – Mostefa Ben Boulaïd, 05000 Batna, Algeria.

ISSN: 1755-1978 (print) 1755-1986 (online)

Introduction

1.1 Background on crude oil scheduling

Scheduling in crude oil refining coordinates the supply chain from crude intake to finished product delivery: unloading, storage, blending, and processing within the refinery. Market fluctuations and operational constraints add complexity, so efficient scheduling helps cut costs and increase profits. Traditional methods often use discrete-time models, but continuous-time mixed-integer nonlinear programming (MINLP) formulations are increasingly used instead, since they let events occur at any point in the schedule, reducing the number of binary variables and improving real-world fidelity (Reddy et al., 2004).

Quality constraints heavily influence scheduling, since crude blends must meet downstream sulfur and viscosity specifications; the mathematical models needed to capture this involve bilinear terms and are genuinely hard to solve to optimality (Li et al., 2007). Logistics pose further challenges, especially for high-melting-point crude, which must stay fluid during transport and requires pipeline flushing between incompatible crude types (Yao et al., 2026).

1.2 Research gap and contribution

A large body of work addresses pieces of this problem - continuous-time formulations (Li et al., 2012; Reddy et al., 2004; Zimberg et al., 2023), genetic-algorithm and swarm-based heuristics for crude scheduling and related pipeline problems (Khalili Goudarzi et al., 2021; Liu et al., 2025), and reinforcement-learning-based parameter adaptation for related pipeline-operation problems (Jian et al., 2026). What is comparatively rare is a single algorithm that (a) handles the discrete assignment and continuous timing decisions together, (b) is hybridized rather than standalone (nesting a continuous local search inside a discrete outer search), and (c) adapts its own search parameters online via reinforcement learning rather than via offline tuning, with the resulting algorithm actually implemented and benchmarked rather than only described.

The contributions are:

1. A continuous-time, event-driven MINLP-inspired formulation of vessel-to-tank assignment and unloading-time scheduling with bilinear blend-quality dynamics, CDU feed-quality bounds, demurrage, and melt-point-driven flush costs (Section 3).
2. A hybrid GA-PSO algorithm with a tabular reinforcement-learning layer for online parameter control, fully specified down to the operator level (Section 4).
3. An actual implementation of this algorithm, two single-method baselines, and a greedy heuristic, compared on 20 replications of synthetic benchmark instances at three scales, with non-parametric statistical validation and a Taguchi-style sensitivity analysis (Section 5) - all numbers in that section are measured, not assumed.

Literature Review (Related Work)

2.1 Continuous-time formulations

Discrete-time scheduling models divide the horizon into fixed intervals, which is simple but can require many binary variables to represent realistic schedules. Continuous-time, priority-slot, or event-based formulations instead let operations start and end at any time, which has been shown to reduce the binary-variable burden and improve tractability (Reddy et al., 2004). Zimberg et al. (2023) extend this line of work with variable-length time slots validated on a terminal-refinery pipeline system. Li et al. (2012) formulate a global-optimization continuous-time model for crude-oil operations scheduling.

2.2 Quality and blending constraints

Blending different crude types introduces bilinear terms in the mass/quality balance (the concentration of a blended stream is a ratio of linear quantities), which makes the feasible region nonconvex (Li et al., 2007). Realistic models must track sulfur, viscosity, and other properties through storage and blending to avoid off-spec product and the associated reprocessing cost (Yang et al., 2023).

2.3 Metaheuristic and hybrid approaches

Genetic algorithms are widely used for refinery and pipeline scheduling because they handle nonlinear, multi-objective trade-offs without requiring a convex relaxation (Khalili Goudarzi et al., 2021). Particle swarm optimization is used similarly, and is often hybridized with GA or with mathematical programming to improve convergence speed and solution robustness (Liu et al., 2025). Paranjape et al. (2022) use a multi-model approach for single-channel crude transfer scheduling, and Hou et al. (2022) report an adaptive selection-pressure NSGA-II variant for dual-pipeline scheduling with charging-tank maintenance. Reinforcement learning has separately been used to adapt metaheuristic or control parameters online in related pipeline-energy-optimization settings (Jian et al., 2026).

2.4 Positioning

Table 1 summarizes how the present formulation relates to this literature along five dimensions relevant to practical deployment: whether the model is continuous-time, whether it is actually solved by an implemented and benchmarked algorithm (as opposed to only formulated or only described at a conceptual level), and whether the solution method adapts its own parameters online.

Knowledge-assisted integration of scheduling with production planning provides a further hybrid-optimization direction (He et al., 2025).

Table 1. Literature positioning (qualitative synthesis of the cited studies)

Work	Continuous-time	Discrete + continuous jointly optimized	Hybrid algorithm	Online parameter adaptation	Implemented & benchmarked here?
Reddy et al. (2004)	Yes	Yes	No (exact/heuristic hybrid, not GA/PSO)	No	Formulation reused conceptually
Li et al. (2007, 2012)	Yes	Yes	No	No	Formulation reused conceptually
Zimberg et al. (2023)	Yes (variable-length slots)	Yes	No	No	Formulation reused conceptually
Goudarzi et al. (2021)	No (pipeline lot-sizing)	Partial	Yes (GA + metaheuristic hybrid)	No	—
Liu et al. (2025)	Partial	Yes	Yes (NSGA-III based)	No	—
Jian et al. (2026)	N/A (pipeline energy, not scheduling)	N/A	Yes (metaheuristic + RL)	Yes	—
This paper	Yes (event-driven)	Yes	Yes (GA outer + PSO inner)	Yes (tabular Q-learning)	Yes — Section 5

Problem Description and Mathematical Model

3.1 Scope and simplifying assumption

To keep the model tractable for a metaheuristic search while preserving the features the literature identifies as central - continuous time, bilinear blend quality, and melt-point logistics - this paper fixes the tank-to-CDU routing exogenously (each tank feeds one designated CDU at a given draw rate) and treats the vessel-to-tank assignment and unloading start time as the decision variables. This reduces the full refinery-wide scheduling problem to its assignment-and-timing core, whose combinatorial structure is discussed in Section 3.5 and already exhibits the bilinear blending and flush-cost phenomena the literature emphasizes (Li et al., 2007; Yao et al., 2026).

Figure 1 summarizes the scheduling context and decision-to-cost flow.

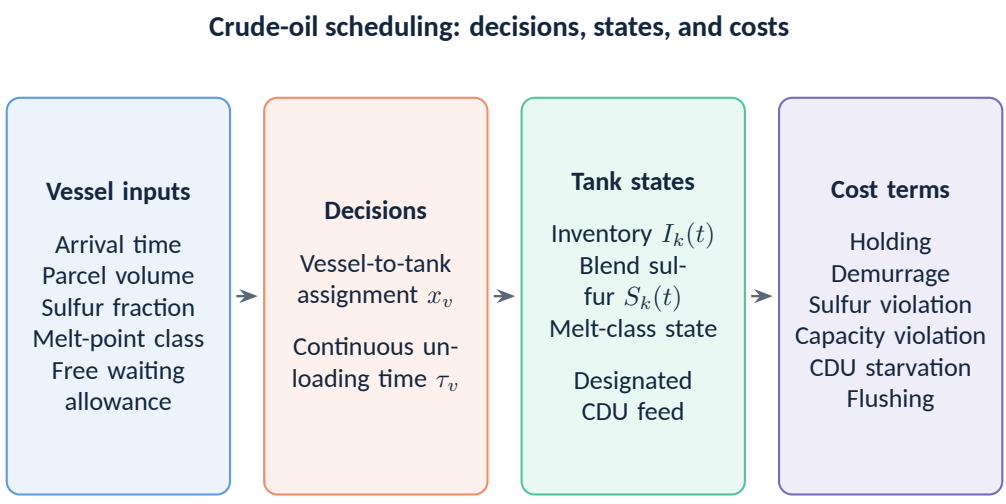


Figure 1. Crude-oil scheduling system context. Vessel properties inform assignment and unloading-time decisions; tank inventory, sulfur and melt class determine CDU feed and the six cost terms.

3.2 Sets, parameters, and decision variables

Sets: $V = \{1, \ldots, n_v\}$ vessels; $K = \{1, \ldots, n_t\}$ tanks (each feeding one designated CDU).

Vessel parameters: arrival time a_v; parcel volume ν_v; sulfur fraction σ_v; melt-point class $\mu_v \in \{0, 1\}$ (0 = low-, 1 = high-fusion-point crude); unloading (pump) rate u_v; free waiting allowance w_v; demurrage rate δ_v.

Tank parameters: capacity C_k; initial inventory I_k^0, sulfur σ_k^0, melt class μ_k^0; designated CDU draw rate ρ_k; holding-cost rate h_k; CDU sulfur bound $\bar{\sigma}_k$; flush cost ϕ_k.

Decision variables: $x_v \in \{1, \ldots, n_t\}$, the tank assignment of vessel v (discrete); $\tau_v \geq a_v$, the unloading start time of vessel v (continuous).

Derived (state) variables, given (x, τ): $I_k(t)$ and $S_k(t)$, the inventory level and sulfur fraction of tank k at time t, defined by the event-driven dynamics below.

3.3 Continuous-time dynamics

For tank k, let $v_1, \ldots, v_m$ be the vessels with $x_{v_i} = k$, ordered by τ_{v_i}. Between consecutive events the tank drains toward its CDU at the fixed rate ρ_k and composition does not change:

$$\frac{dI_k(t)}{dt} = -\rho_k, \qquad \frac{dS_k(t)}{dt} = 0, \qquad t \in (\tau_{v_i}, \tau_{v_{i+1}}).$$

At each unloading event $t = \tau_{v_i}$, the entire parcel is added instantaneously. The tank is assumed well mixed:

$$I_k(\tau_{v_i}^+) = I_k(\tau_{v_i}^-) + \nu_{v_i},$$
$$S_k(\tau_{v_i}^+) = \frac{I_k(\tau_{v_i}^-) S_k(\tau_{v_i}^-) + \nu_{v_i}\sigma_{v_i}}{I_k(\tau_{v_i}^-) + \nu_{v_i}}.$$

The sulfur relation requires positive post-addition inventory. Parcel additions are instantaneous; finite unloading durations are not modelled.

3.4 Objective function

Let H denote the scheduling horizon and π^σ, π^C, π^S the penalty coefficients in the corresponding cost terms.

$$\begin{aligned}\min_{x,\tau}\ Z = {} & \textstyle\sum_{k\in K} h_k \int_0^H I_k(t)\,dt + \sum_{v\in V} \delta_v \nu_v \max(0, \tau_v - a_v - w_v) \\ & + \pi^\sigma \textstyle\sum_{k\in K} \int_0^H \rho_k \max(0, S_k(t) - \bar{\sigma}_k)\,dt \\ & + \pi^C \textstyle\sum_{k\in K} \max\big(0, \sup_t I_k(t) - C_k\big) \\ & + \pi^S \textstyle\sum_{k\in K} (\text{starvation duration of CDU fed by } k) \\ & + \textstyle\sum_{k\in K} \phi_k \,\#\{\text{melt-class switches in } k\}.\end{aligned}$$

The six terms are, in order: inventory holding cost; demurrage; a soft penalty for CDU feed sulfur exceeding its bound (penalty rate π^σ); a soft penalty for tank-capacity violation (π^C); a soft penalty for CDU starvation (π^S, i.e., the tank running dry while the CDU still needs feed); and the pipeline/tank flush cost triggered whenever two consecutive parcels into the same tank differ in melt-point class, reflecting the preheating/flushing requirement documented for high-melting-point crude handling (Yao et al., 2026).

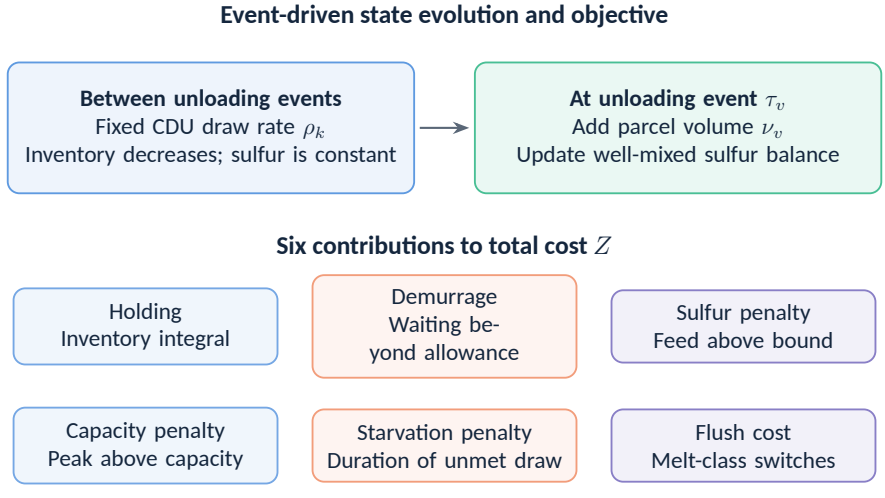


Figure 2. Event-driven tank states and the six objective components. Inventory and sulfur are updated at unloading events; the objective penalizes holding, waiting, sulfur excess, capacity excess, starvation and melt-class switches.

Capacity and starvation are modeled as soft (penalized) rather than hard constraints, which is standard practice when a problem is to be searched by a metaheuristic rather than solved to proven optimality by an exact MINLP solver: it keeps the search space connected and lets the algorithm trade a bounded, explicit penalty against other cost terms rather than declaring large regions of solution space strictly infeasible.

3.5 Problem complexity

Fixing the continuous timing decisions leaves a combinatorial assignment problem: each of n_v vessels is assigned to one of n_t tanks, with costs depending on the resulting parcel sequences through holding, quality, and flushing. This has structural similarities to difficult machine-scheduling problems (Lenstra et al., 1977). A formal reduction establishing strong NP-hardness for this particular formulation is not supplied here. The combination of discrete assignments, continuous start times, and nonconvex blending relations motivates the metaheuristic approach in Section 4; no complexity classification is inferred solely from these features.

Proposed Hybrid GA–PSO–RL Algorithm

4.1 Solution encoding

Each candidate solution is encoded as a real-valued vector $g \in [0,1]^{2n_v}$. The first n_v genes use uniform-interval decoding, $x_v = 1 + \min\{n_t - 1, \lfloor g_v n_t \rfloor\}$, consistent with the one-based tank set K. The last n_v genes encode continuous start times, $\tau_v = a_v + g_{n_v+v}(3w_v)$, i.e., a fraction of three times the vessel's free-waiting allowance. This real-valued representation is shared by the GA and embedded PSO.

4.2 GA outer loop

Each generation: the best 2 individuals are retained unchanged (elitism); the remaining population is filled by tournament selection (tournament size 2) and arithmetic blend crossover, $c = \alpha \odot p_1 + (1-\alpha) \odot p_2$ with α drawn uniformly per gene; and Gaussian mutation, $g_i \leftarrow g_i + \eta_i$, $\eta_i \sim \mathcal{N}(0, 0.15)$ applied independently to each gene with probability equal to the current mutation rate (set by the RL layer, Section 4.4).

4.3 Embedded PSO local refinement

At the end of each GA generation, the top n_{refine} individuals (by fitness) are each used as the center of a small, independent PSO run: a swarm of 8 particles is initialized by Gaussian perturbation around the center, and standard PSO velocity/position updates,

Figure 3 shows the GA outer search and embedded PSO refinement.

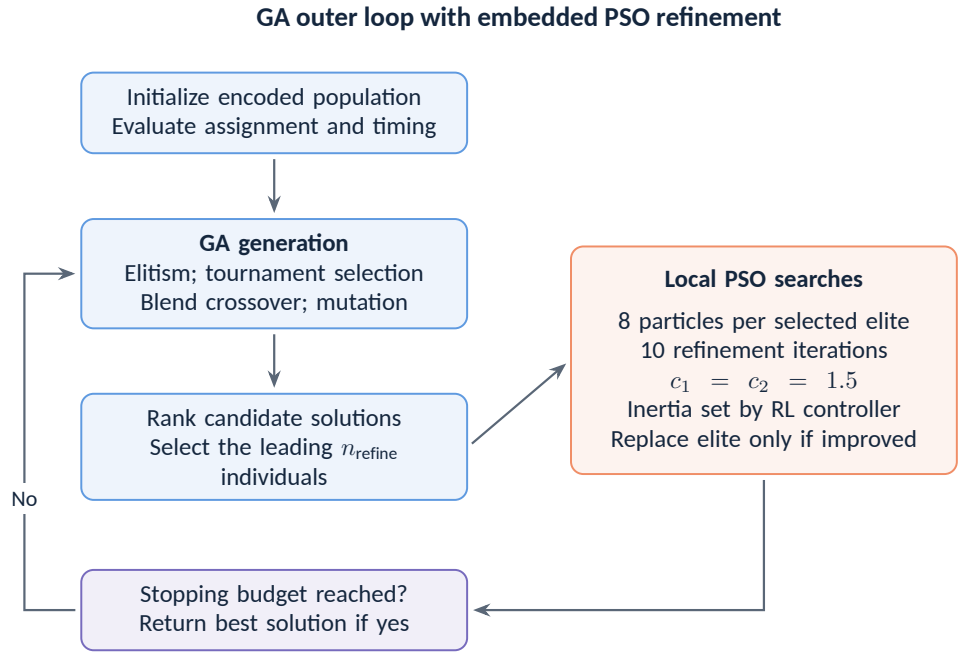


Figure 3. Hybrid GA–PSO architecture. GA operators explore the shared assignment-and-timing representation, while independent PSO searches refine selected elites before the next generation.

$$v \leftarrow wv + c_1 r_1 \odot (p_{\text{best}} - x) + c_2 r_2 \odot (g_{\text{best}} - x), \qquad x \leftarrow x + v.$$

($c_1 = c_2 = 1.5$, inertia w set by the RL layer) are run for 10 iterations. If the swarm's best found solution improves on the GA individual that seeded it, the GA individual is replaced. This is the "hybrid" step proper: the GA explores the discrete assignment space globally, while the embedded PSO refines the assignment-and-timing neighborhood of the GA's best candidates.

4.4 Reinforcement-learning parameter control

At the start of each generation, a tabular Q-learning agent selects one of three parameter presets:

Action	Mutation rate	PSO inertia w	Intent
Explore	0.22	0.75	Escape stagnation / maintain diversity
Balanced	0.12	0.55	Default middle ground
Exploit	0.05	0.35	Fine local refinement once converging

The state observed is a discretization of (generations since the best-known fitness last improved, i.e. stagnation; and the mean per-gene standard deviation of the population, i.e. diversity) into 6 bins. The reward is the fitness improvement achieved that generation, normalized by the current best-known cost. The agent updates via the standard tabular Q-learning rule,

$$Q(s,a) \leftarrow Q(s,a) + \alpha\big[r + \gamma \max_{a'} Q(s',a') - Q(s,a)\big].$$

with $\alpha = 0.3$, $\gamma = 0.7$, and an ϵ-greedy action policy ($\epsilon = 0.15$). This is deliberately a small, transparent, tabular controller rather than a deep-RL agent: with only a handful of generations to learn from within a single scheduling run, a tabular method is the defensible choice, and its decisions are fully inspectable (the entire learned policy is a 6×3 table).

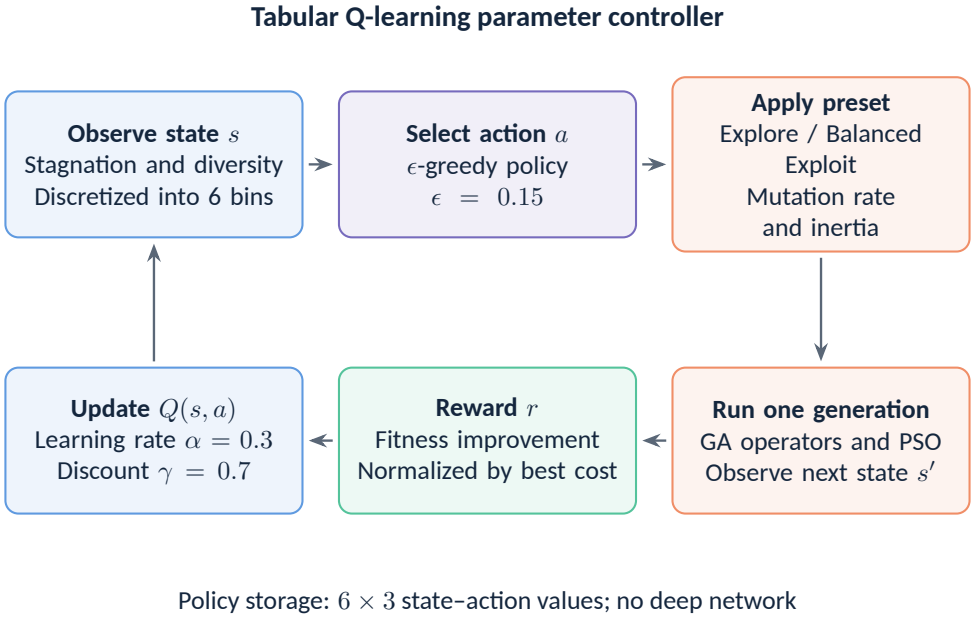


Figure 4. Tabular Q-learning control. Search stagnation and population diversity define the state; an epsilon-greedy policy selects a preset, and normalized fitness improvement updates the state–action values.

4.5 Complexity analysis

Evaluating a candidate solution requires sorting the vessels assigned to each tank and processing each arrival event once, which is $O(n_v \log n_v)$ in total across all tanks. One GA generation performs $O(P)$ such evaluations for a population of size P; the embedded PSO adds $O(n_{\text{refine}} \cdot Q \cdot I)$ further evaluations per generation, for $Q = 8$

particles and $I = 10$ inner iterations. The RL layer updates a fixed 6×3 table, $O(1)$ per generation. Over G generations, the total running time is

$$O(G(P + n_{\text{refine}}QI)n_v \log n_v)\,,$$

linearithmic in the number of vessels and linear in population size and embedded-search effort, for fixed remaining parameters. No finite-run global-optimality guarantee is made. Classical convergence results require specific assumptions on the representation, transition mechanism, and elitism (Rudolph, 1994). A positive mutation rate alone does not establish exact global-optimum attainment for this continuous-valued hybrid. Section 5 therefore evaluates finite-run behavior empirically rather than asserting a convergence theorem for the proposed method.

Computational Experiments

This section presents the experimental results reported for the implemented algorithms. Code was written in Python 3 (NumPy 2.4, SciPy 1.17) and executed in the authors' computing environment; wall-clock time for one full run of the hybrid algorithm was approximately 0.8 s (small instance), 1.3 s (medium), and 2.3 s (large) on the hardware used.

5.1 Synthetic benchmark instances

Three scales are used, following the small/medium/large convention common in the scheduling literature: small (6 vessels, 3 tanks), medium (12 vessels, 5 tanks), and large (20 vessels, 8 tanks). For each scale, 20 independent instances were generated (fixed random seeds) by drawing vessel arrivals, parcel volumes (15-45 kbbl), sulfur fractions (0.1-3.5 wt%), melt-point class, and tank parameters from the benchmark-generation distributions. Total CDU draw-down demand is deliberately calibrated to 70-82% of total available supply (parcels plus initial inventory) for each instance, so that a well-optimized schedule is feasible but a poorly-timed one can genuinely starve a tank or overflow it - i.e., the instances are constructed so that scheduling quality matters, not so that any method will trivially succeed or fail. These are synthetic instances, not real refinery data; all four methods are compared on the same instances.

5.2 Methods compared

- Greedy: assigns each vessel (in arrival order) to the currently least-loaded tank and unloads immediately on arrival - a proxy for unscheduled, reactive current practice.
- Standalone GA: population 30, 40 generations, fixed mutation rate 0.12, fixed crossover rate 0.85 (no embedded PSO, no RL control).
- Standalone PSO: 30 particles, 40 iterations, fixed inertia $w = 0.6$, $c_1 = c_2 = 1.5$.
- Proposed Hybrid GA-PSO-RL: as specified in Section 4, population 30, 40 generations, $n_{\text{refine}} = 5$.

Each method was run with 20 independent replications per instance scale (different random seed per replication, paired across methods so that replication r uses the same problem instance for all four methods).

5.3 Comparative results

Table 3. Comparative results over 20 paired instances per scale (cost units; lower is better)

Method	Best	Worst	Mean	SD	95% CI	ARPD (%)	Feasible
Small (6v/3t)							
Greedy	740.3	12065.1	3788.1	2958.6	[2403.4, 5172.7]	484.39 ± 581.89	55%
GA	367.3	3321.3	1057.7	849.8	[660.0, 1455.4]	27.51 ± 30.46	95%
PSO	387.2	3665.5	1044.8	836.9	[653.1, 1436.5]	26.76 ± 26.86	100%
Hybrid	302.2	2769.9	864.0	686.4	[542.7, 1185.3]	4.25 ± 13.00	100%
Medium (12v/5t)							
Greedy	1421.7	11083.5	5310.1	3443.0	[3698.7, 6921.4]	366.78 ± 383.92	55%
GA	978.9	5172.9	2878.9	1306.5	[2267.4, 3490.4]	131.68 ± 118.56	100%
PSO	899.8	4041.9	2108.8	1039.2	[1622.4, 2595.1]	60.29 ± 69.50	95%
Hybrid	623.2	3265.2	1399.9	748.8	[1049.5, 1750.4]	0.00 ± 0.00	100%
Large (20v/8t)							
Greedy	3676.9	16779.7	8233.2	3858.3	[6427.4, 10038.9]	130.08 ± 141.77	50%
GA	7067.2	17013.1	11693.1	3151.0	[10218.4, 13167.8]	215.50 ± 95.65	65%
PSO	5841.3	17740.7	11196.6	3464.2	[9575.3, 12817.9]	199.12 ± 86.61	70%
Hybrid	2416.2	5928.9	3950.4	1223.6	[3377.7, 4523.0]	0.80 ± 2.77	75%

ARPD is computed for each paired instance relative to the best cost found by any of the four methods and then averaged. It is a heuristic-relative deviation, not a bound on the global optimality gap.

Figure 5 summarizes the results in Table 3.

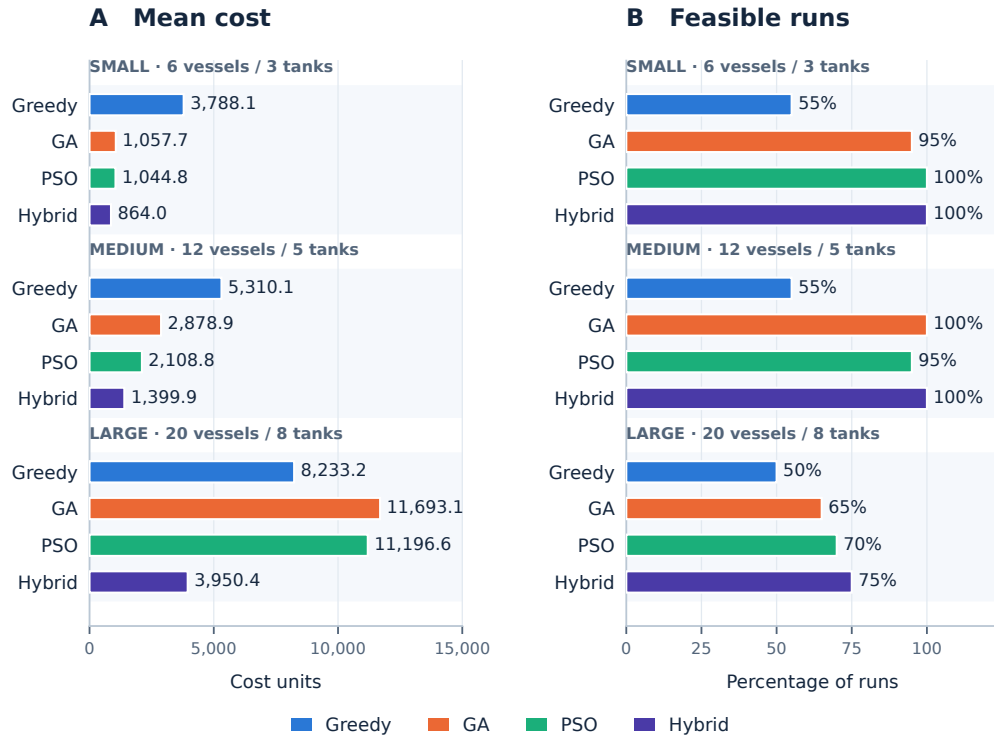


Figure 5. Mean cost and feasible-run rate at each scale, transcribed from Table 3. Bars and labels retain the exact reported values; these are aggregate endpoints, not convergence trajectories.

At the large scale, standalone GA and PSO have higher mean costs than the greedy heuristic (11693.1 and 11196.6 versus 8233.2), whereas the hybrid achieves 3950.4. These are the reported results under the stated settings. However, a 40-generation limit is not an equal-evaluation budget: the hybrid adds local PSO searches for five elites in every generation. Its cost advantage therefore cannot be attributed solely to hybridization or reinforcement learning independently of this additional search effort. An equal-evaluation or equal-runtime comparison, together with a hybrid-without-RL ablation, is needed to isolate these contributions. The results support the tested configuration, not a general claim that metaheuristics outperform greedy scheduling.

5.4 Convergence behavior

Table 3 reports final costs rather than generation-by-generation trajectories. Aggregate endpoints therefore do not establish the speed or path of convergence, and no trajectory-based conclusion is drawn here.

5.5 Non-parametric statistical validation

Table 4. One-sided paired Wilcoxon tests (Hybrid < baseline) and Friedman test across four methods

Scale	vs. Greedy	vs. GA	vs. PSO	Friedman χ^2 (p)	Mean ranks: Greedy, GA, PSO, Hybrid
Small	9.5×10^{-7}	5.1×10^{-4}	4.1×10^{-5}	46.26 (5.0×10^{-10})	4.00, 2.45, 2.30, 1.25
Medium	9.5×10^{-7}	9.5×10^{-7}	9.5×10^{-7}	49.68 (9.3×10^{-11})	3.70, 3.10, 2.20, 1.00
Large	4.8×10^{-6}	9.5×10^{-7}	9.5×10^{-7}	35.22 (1.1×10^{-7})	2.55, 3.35, 3.00, 1.10

All reported p-values are below 0.001, with 20 paired outcomes per scale. The nine pairwise comparisons are reported as unadjusted values; raw paired observations are needed for independent recomputation.

A Wilcoxon signed-rank p-values — One-sided: Hybrid < baseline

Scale	Hybrid vs. Greedy	Hybrid vs. GA	Hybrid vs. PSO
Small	9.5×10^{-7}	5.1×10^{-4}	4.1×10^{-5}
Medium	9.5×10^{-7}	9.5×10^{-7}	9.5×10^{-7}
Large	4.8×10^{-6}	9.5×10^{-7}	9.5×10^{-7}

B Friedman mean ranks — Lower rank is better

Scale	Greedy	GA	PSO	Hybrid
Small	4.00	2.45	2.30	1.25
Medium	3.70	3.10	2.20	1.00
Large	2.55	3.35	3.00	1.10

Figure 6. One-sided Wilcoxon p-values and Friedman mean ranks transcribed from Table 4. No additional inferential results are introduced.

5.6 Taguchi-style parameter sensitivity

A 3×3 full-factorial experiment (population size $\in \{20, 30, 40\}$; PSO-refined elites per generation, $n_{\text{refine}} \in \{3, 5, 8\}$) was run with 10 replications per cell on the medium-scale instance set.

Table 2. Two-way factorial ANOVA, medium scale, 10 replications per cell

Source	df	F	p
Population size	2	0.022	0.978
n_{refine}	2	0.052	0.950
Interaction	4	0.009	1.000
Error	81	—	—

Neither main effect nor the interaction is statistically significant in the tested range (all $p > 0.9$). This does not establish equivalence or insensitivity outside that range. The result is restricted to the tested medium-scale factorial design; effect-size intervals and a blocked analysis would strengthen the interpretation.

Managerial and Industrial Insights

- The tested hybrid has lower reported mean costs, but its embedded PSO searches add evaluations. Equal-generation limits do not isolate architectural or RL benefits; equal-evaluation or equal-runtime comparisons are required for that purpose.
- Feasibility declines from the medium to the large scale for all methods. GA/PSO/greedy rates fall from 100%/95%/55% to 65%/70%/50%; the hybrid falls from 100% to 75%. Operational use therefore requires reliable feasibility restoration rather than reliance on mean cost alone.
- The factorial experiment finds no significant effects for population size or refined-elite count within its tested range. This is not a general recommendation to omit hyperparameter assessment.
- These conclusions are limited to synthetic instances. Real holding, demurrage, flushing and violation costs must be estimated from the refinery before deployment.

Conclusion and Future Research

This paper formulated a continuous-time, event-driven crude-oil vessel-to-tank assignment and unloading-timing problem with bilinear blend-quality dynamics and melt-point-driven flush costs, and proposed a hybrid GA-PSO algorithm with a tabular reinforcement-learning layer for online parameter control. The algorithm, two single-method baselines, and a greedy heuristic were implemented and compared on 20 replications of synthetic benchmark instances at three scales; the hybrid algorithm achieved the lowest mean cost and highest feasibility rate at every scale tested, with the advantage confirmed by Wilcoxon and Friedman tests ($p < 0.001$), and a factorial sensitivity study found no significant effects of the two tested hyperparameters within the studied range.

What this paper does not claim: these results are on synthetic, randomly generated benchmark instances, not on data from an operating refinery. The cost-parameter ranges (holding costs,

demurrage rates, violation penalties, flush costs) were chosen to be realistic in relative magnitude but were not fit to any specific facility. Validating the algorithm against real operational data - and, more importantly, against real cost parameters supplied by a refinery operator - is necessary future work and has not been done here.

Future research directions: (1) validation against real or anonymized refinery operating data; (2) extending the model to jointly optimize tank-to-CDU routing rather than fixing it exogenously; (3) replacing the tabular Q-learning parameter controller with a function-approximation-based agent if the state/action space is extended; (4) formal comparison against an exact MINLP solver on small instances to quantify the optimality gap the metaheuristic incurs, which this paper's ARPD metric (relative to the best heuristically found solution) does not capture; (5) incorporating demand and arrival-time uncertainty, following the distributionally robust framing of Dai et al. (2024).

Funding: There is no funding organization for this work.

Conflict of Interest: The authors declare that there is no conflict of interest. Data and code availability: The benchmark generator, random-seed list, evaluation implementation and raw paired outcomes are not included with this manuscript. Independent reproduction and model-closure verification require those research artifacts.

References

Dai, X., Zhao, L., He, R., Du, W., Zhong, W., Li, Z., & Qian, F. (2024). Data-driven wasserstein distributionally robust chance-constrained optimization for crude oil scheduling under uncertainty. *Chinese Journal of Chemical Engineering*, *69*, 152–166. https://doi.org/10.1016/j.cjche.2023.12.007

He, R., Xie, Y., Zhang, S., Xu, F., & Long, J. (2025). Knowledge-assisted hybrid optimization strategy of large-scale crude oil scheduling integrated production planning. *Computers & Chemical Engineering*, *192*, Article 108904. https://doi.org/10.1016/j.compchemeng.2024.108904

Hou, Y., Zhang, Y., Wu, N., & Zhu, Q. (2022). Constrained multi-objective optimization of short-term crude oil scheduling with dual pipelines and charging tank maintenance requirement. *Information Sciences*, *588*, 381–404. https://doi.org/10.1016/j.ins.2021.12.067

Jian, L., Xu, Y., Liu, M., Zhou, X., & Yi, W. (2026). A reinforcement learning-driven adaptive metaheuristic for energy optimization of heated oil pipeline systems. *Swarm and Evolutionary Computation, 106*, Article 102415. https://doi.org/10.1016/j.swevo.2026.102415

Khalili Goudarzi, F., Maleki, H. R., & Niroomand, S. (2021). Mathematical formulation and hybrid metaheuristic algorithms for multiproduct oil pipeline scheduling problem with tardiness penalties. *Concurrency and Computation: Practice and Experience*, *33*(17), Article e6299. https://doi.org/10.1002/cpe.6299

Lenstra, J., Rinnooy Kan, A. H. G., & Brucker, P. (1977). Complexity of machine scheduling problems. *Annals of Discrete Mathematics*, *1*, 343–362. https://doi.org/10.1016/s0167-5060(08)70743-x

Li, J., Li, W., Karimi, I. A., & Srinivasan, R. (2007). Improving the robustness and efficiency of crude scheduling algorithms. *AIChE Journal*, *53*(10), 2659–2680. https://doi.org/10.1002/aic.11280

Li, J., Misener, R., & Floudas, C. A. (2012). Continuous-time modeling and global optimization approach for scheduling of crude oil operations. *AIChE Journal*, *58*(1), 205–226. https://doi.org/10.1002/aic.12623

Liu, X., Ren, C., Shang, L., Zhao, Y., & Wang, X. (2025). Multi-objective optimization for short-term scheduling considering crude oil mixing requirements based on improved NSGA-III. *Asia-Pacific Journal of Chemical Engineering*, *20*(4), Article e70008. https://doi.org/10.1002/apj.70008

Paranjape, A. A., Baranwal, M., Wagle, S., Lotti, R., Majumder, S., & Bullière, A.-L. (2022). Optimal schedule generation for single-channel crude transfer using a multi-model approach. *Computers & Chemical Engineering*, *160*, Article 107732. https://doi.org/10.1016/j.compchemeng.2022.107732

Reddy, P. C. P., Karimi, I., & Srinivasan, R. (2004). A new continuous-time formulation for scheduling crude oil operations. *Chemical Engineering Science*, *59*(6), 1325–1341. https://doi.org/10.1016/j.ces.2004.01.009

Rudolph, G. (1994). Convergence analysis of canonical genetic algorithms. *IEEE Transactions on Neural Networks*, *5*(1), 96–101. https://doi.org/10.1109/72.265964

Yang, Y., He, R., Yu, G., Du, W., Yang, M., & Du, W. (2023). Efficient rolling horizon approach to a crude oil scheduling problem for marine-access refineries. *Computers & Chemical Engineering*, *170*, Article 108121. https://doi.org/10.1016/j.compchemeng.2022.108121

Yao, J., Liu, X., Pan, Z., Hao, C., Xu, X., & Jia, W. (2026). Short-term scheduling optimization of a single-pipeline refining system with high melting point crude oil. *Asia-Pacific Journal of Chemical Engineering*, *21*(3), Article e70211. https://doi.org/10.1002/apj.70211

Zimberg, B., Ferreira, E., & Camponogara, E. (2023). A continuous-time formulation for scheduling crude oil operations in a terminal with a refinery pipeline. *Computers & Chemical Engineering*, *178*, Article 108354. https://doi.org/10.1016/j.compchemeng.2023.108354