



Research article

A reinforcement learning-based rolling optimization method for weekly aircraft maintenance scheduling under repair efficiency uncertainty

Shihong Li^{1,2}, Pengliang Che¹, Mengjin Qu¹ and Qing Li^{1,*}

¹ Department of Automation, Tsinghua University, Beijing 100083, China

² Aviation University of Air Force, Changchun 130022, China

* **Correspondence:** Email: liqing@tsinghua.edu.cn.

Abstract: Weekly aircraft maintenance scheduling is a path-dependent sequential decision problem in which dynamic task arrivals, inherited backlog, and uncertain repair efficiency reshape future maintenance capacity. We propose a feasibility-aware proximal policy optimization (PPO) rolling framework that learns aircraft-level task allocation and proactive maintenance preferences and converts them into executable weekly schedules through deterministic priority conversion, eligibility logic, capacity projection, integerization, and a common physical executor. A 50-week evaluation with 10 paired seeds compared the method with FCFS, Greedy-Urgency, an exact one-week receding-horizon MILP, and a genetic algorithm rolling heuristic. In the $N=20$ base case, the proposed PPO improved task pool service by 2.05 percentage points and reduced average backlog by 86.89 relative to Greedy-Urgency; relative to FCFS, it increased aircraft availability by 11.09 percentage points, reduced cost by 258.60, and lowered constraint pressure by 34.60 percentage points. It also achieved higher long-horizon service and lower backlog than the exact MILP and GA baselines with approximately 5 ms online decision time. Component studies showed that proactive maintenance improves long-horizon service and backlog control, while action masking improves availability, cost, constraint pressure, and proposal feasibility. The architecture maintained a consistent operating profile across repair efficiency variation and fleet sizes $N=10\text{--}30$, with online decision times of 3.463–7.173 ms. The resulting framework provides transparent and computationally efficient decision support for weekly aircraft maintenance under uncertain execution capacity.

Keywords: aircraft maintenance scheduling; rolling optimization; repair efficiency uncertainty; feasibility projection; proactive maintenance; PPO

Mathematics Subject Classification: 90B06; 90B35; 68T05

1. Introduction

Aircraft maintenance is central to airworthiness, operational continuity, and fleet availability. Maintenance decisions directly influence flight schedules, mission capability, turnaround efficiency, and operating cost. Research on condition-based fleet maintenance, predictive maintenance, maintenance check scheduling, and line maintenance scheduling demonstrates the operational value of coordinating maintenance timing with aircraft state and resource availability [1–3]. In high-density and resource-constrained operations, poorly coordinated maintenance can propagate into overdue tasks, aircraft unavailability, resource rescheduling, and downstream operational disruption. Aircraft maintenance scheduling has therefore evolved into an integrated optimization problem that couples task priority, resource capacity, aircraft state, and long-horizon operational performance[4].

Aircraft maintenance policies span preventive, corrective, condition-based, and predictive strategies. Preventive maintenance is scheduled by flight hours, flight cycles, or calendar time, which supports planning but may consume component life and maintenance capacity prematurely. Corrective maintenance exploits component life more fully but can create unscheduled downtime and emergency workload. Condition-based and predictive maintenance use health monitoring, remaining useful life (RUL) prediction, and state assessment to align maintenance timing with fleet condition [5–7]. Related studies address multi-component spare constraints, engine-level alarm decisions, and adaptive RUL prediction [8–10]. Reinforcement learning has increasingly complemented these approaches by supporting sequential maintenance decisions in dynamic and uncertain environments [11–13].

Weekly maintenance introduces three coupled features that are less prominent in long-horizon check planning or local line maintenance allocation. First, the task pool evolves continuously as new tasks arrive, and unfinished work is inherited from previous weeks. Second, realized repair efficiency changes the amount of work that can be executed within a weekly capacity. Third, proactive maintenance can consume current resources while changing future task pressure and flight-hour margins. These features make weekly maintenance a rolling optimization problem in which backlog propagation, execution uncertainty, and proactive decisions must be coordinated over time[14,15].

These couplings create pronounced state and path dependence. Incomplete work enlarges the next week's task pool; lower repair efficiency reduces executable capacity; and the timing of proactive maintenance changes both near-term availability and future maintenance pressure. A locally attractive weekly action can therefore reshape later fleet states, making horizon-aware decision-making essential. Reinforcement learning is well-suited to this setting because it can learn state-feedback decisions from the cumulative consequences of repeated weekly interactions, while an explicit execution layer preserves the operational constraints of the maintenance process[16–18].

To address this problem, we develop an executable rolling architecture that integrates a learned state-feedback policy, deterministic priority-to-schedule conversion, feasibility-aware projection, and a common physical executor. The contribution lies in coupling these elements within a weekly state-transition process in which current decisions shape future backlog, executable capacity, and aircraft availability. This architecture converts learned priorities into auditable schedules and supports direct comparison with rule-based, exact, evolutionary, and mechanism-ablated alternatives under common execution semantics.

The main contributions are summarized as follows:

(1) We formulate weekly aircraft maintenance as an executable rolling state-transition model that jointly represents dynamic task arrivals, inherited backlog, aircraft states, flight-hour margins, repair efficiency-dependent execution capacity, and proactive maintenance. The evaluation semantics are explicit: TCR measures rolling task pool service, and SVR measures model-internal scheduling constraint pressure.

(2) We construct a feasibility-aware PPO scheduling mechanism in which the actor learns aircraft-level latent task allocation and proactive maintenance preferences, while deterministic priority conversion, eligibility logic, capacity projection, integerization, and the common executor generate the weekly schedule. PPO likelihood is evaluated on the sampled latent action, preserving consistency between rollout sampling and policy updating while keeping execution feasibility explicit.

(3) We benchmark the rolling mechanism over 50-week simulations against FCFS, Greedy-Urgency, an exact one-week receding-horizon MILP, and a genetic algorithm rolling heuristic. Targeted component studies quantify the intertemporal value of proactive maintenance and the feasibility contribution of action masking, with a dedicated information-sensitivity study for direct repair efficiency observation.

(4) We characterize the operating envelope across repair efficiency intensity, repair efficiency distribution family, demand intensity, and fleet sizes $N=10$, 20, and 30. The experiments show stable multi-objective behavior across repair efficiency variation and the tested fleet size range, together with millisecond-level online computation as the policy scales from $N=10$ to $N=30$.

The remainder of this paper is organized as follows: Section 2 reviews aircraft maintenance, routing and recovery, uncertainty-aware optimization, and learning-based scheduling. Section 3 formulates the weekly rolling problem and metric semantics. Section 4 presents the feasibility-aware PPO scheduling mechanism and executable action conversion. Section 5 describes training, optimization baselines, independent-instance evaluation, component studies, robustness tests, and statistical reporting. Section 6 reports the comparative, mechanism-level, robustness, and fleet size results. Section 7 concludes the study, and Appendix A summarizes the executable synthetic data generation and replay protocol.

2. Literature review and research positioning

2.1. Aircraft maintenance scheduling and operational planning

Aircraft maintenance scheduling has long been studied as part of airline operations planning. Early operations research work established flight-scheduling, maintenance-routing/base-planning, and fleet-assignment structures that later maintenance scheduling models build upon [19,20]. Later studies developed dynamic programming, mixed-integer programming, and simulation-optimization formulations for maintenance checks, hangar scheduling, tail assignment, disruptive task rescheduling, and fleet-level maintenance coordination [21–23].

Aircraft recovery research provides a complementary operational perspective. Column-generation heuristics coordinate disruption recovery with airport capacity constraints and maintenance flexibility [24–26], reinforcement learning supports multi-fleet recovery under airport disruption [27–29], and recent methods combine evolutionary search, reinforcement learning-guided local improvement, and explicit feasibility repair for multi-objective recovery [30]. These studies motivate strong exact,

heuristic, evolutionary, and learning-based comparisons. Their primary decision objects are disrupted flight rotations, leaving recurring weekly maintenance task pools and their backlog dynamics as a distinct scheduling problem.

These models provide strong foundations for long-horizon check planning and deterministic task allocation. Line-maintenance studies further address short ground stops and local task resource feasibility [4,31]. Weekly rolling maintenance extends this setting by combining continuously renewed task pools, inherited backlog, execution-capacity fluctuations, and proactive maintenance within the same state transition. This combination creates a decision horizon in which current-week allocation changes the workload and aircraft states encountered later.

Weekly maintenance, therefore, requires a sequential decision perspective. The task pool is renewed every week, unfinished workload is carried forward, and current decisions reshape future backlog and fleet availability. The problem studied here is formulated accordingly as a rolling state-dependent scheduling process with explicit cross-week propagation.

2.2. *Predictive maintenance and health-aware decision making*

Fleet-level condition-based maintenance formulations combine partial observability or optimization models with health-state information [1,5,32]. Prognostics-oriented studies use remaining useful life estimates and uncertainty to support predictive maintenance and maintenance timing [2,10]. Together, these studies demonstrate the value of converting health information into state-aware maintenance decisions and provide important foundations for proactive maintenance planning.

The present study targets a complementary source of uncertainty: repair execution efficiency. RUL uncertainty primarily shifts when maintenance demand emerges, whereas repair efficiency uncertainty changes how much maintenance workload can be executed within the current weekly cycle. Execution efficiency, therefore, affects task completion, resource use, backlog carryover, and subsequent fleet availability even when the maintenance demand sequence is unchanged.

Proactive maintenance connects health-aware timing with rolling task allocation. Its value depends on whether current resources can absorb the intervention and whether the intervention reduces future workload pressure. The proposed framework, therefore, learns proactive maintenance preferences jointly with weekly task allocation, allowing their intertemporal effect to be evaluated within the same rolling process.

2.3. *Maintenance scheduling under uncertainty*

Aircraft maintenance under uncertainty has been addressed through robust and simulation-optimization formulations [26,27], while broader maintenance research studies have focused on stochastic deterioration, monitoring quality, imperfect inspection, and information value [33–35]. Dynamic maintenance scheduling has also compared reinforcement learning, genetic algorithm simheuristics, and dispatching rules under changing operating conditions [36,37]. Chance-constrained carrier-aircraft hangar scheduling further illustrates how mission uncertainty, multi-skilled resources, and rescheduling can be integrated into an optimization framework [38].

For weekly aircraft maintenance, uncertainty propagates through the task pool. A reduction in effective repair capacity delays current work and enlarges the backlog entering the next week; the enlarged task pool then changes future resource pressure and aircraft availability. Modeling repair

efficiency directly in executable capacity, therefore, captures a cross-week propagation mechanism that extends beyond isolated task-level processing-time perturbations.

The study addresses this mechanism through a weekly repair efficiency multiplier and evaluates low, medium, and high disturbance levels. The analysis focuses on how execution-capacity variation reshapes the service–backlog–availability–cost trade-off as backlog propagates across weeks.

2.4. *Reinforcement learning for dynamic maintenance optimization*

Reinforcement learning is suitable for sequential maintenance decisions because current actions affect future states, and performance is evaluated through cumulative outcomes [15]. Aircraft-specific applications include maintenance-check scheduling, line maintenance, fleet-availability optimization, and fighter-aircraft maintenance [3,13,14]. Related work used reinforcement learning for manufacturing or condition-based maintenance scheduling [39–41], while inspection and maintenance under incomplete information have been addressed by POMDP and constrained deep reinforcement learning formulations [42,43]. Large-scale airline fleet maintenance and carrier-based flight-deck scheduling extend learning-based scheduling to more complex operational settings [44,45].

Recent work also expanded the methodological scale of learning-based scheduling. Distributed hierarchical multi-agent reinforcement learning has been evaluated for airline fleets ranging from 10 to 200 aircraft [44], and graph neural network–assisted deep reinforcement learning has been used for real-time carrier-based aircraft flight-deck scheduling [45]. A recent systematic review surveyed machine learning–driven combinatorial optimization for routing and scheduling under large-scale and dynamic constraints, reinforcing the need to connect learned decisions with explicit optimization structure and strong baselines [46].

Three research needs emerge for weekly aircraft maintenance. First, repeated task pool service requires explicit representation of inherited backlog and cross-week state evolution. Second, maintenance execution capacity should respond directly to repair efficiency variation. Third, a learning-based scheduler must provide a reproducible mapping from continuous policy output to an executable aircraft-level schedule. These needs motivate an architecture in which learning, feasibility handling, and physical execution are separated but coordinated.

The proposed framework uses PPO as the state-feedback learning engine within this executable rolling architecture. Its methodological contribution lies in the weekly state-transition model, deterministic priority-to-schedule conversion, feasibility projection, and joint treatment of proactive maintenance and repair efficiency information.

2.5. *Structured comparisons and research gap*

Table 1 positions the proposed approach relative to representative aircraft maintenance, routing/recovery, uncertainty-aware, exact/heuristic optimization, and learning-based scheduling studies. The distinguishing contribution is an executable weekly architecture that couples backlog propagation, repair efficiency–dependent capacity, feasibility-aware schedule construction, and proactive maintenance, followed by component and robustness studies that reveal how these mechanisms shape the operating trade-off.

Table 1. Structured comparison with representative studies.

Representative study	Problem and method	Relevant scope	Key distinction
Deng et al. [21]	Long-term maintenance-check scheduling; dynamic programming	Due-window and long-horizon check optimization	Check-level planning with predetermined maintenance structure; different from recurring weekly task pool evolution
Liang et al. [28]	Aircraft recovery with maintenance flexibility; column generation heuristic	Airport capacity, disruption recovery, and flexible maintenance	Recovery-centered planning for disrupted flight rotations rather than recurring weekly maintenance workload
Lee et al. [29]	Multi-fleet aircraft recovery; Q-learning and double Q-learning	Learning-based disruption recovery	Flight-recovery action space and disruption response; different from repair capacity-aware weekly maintenance
Tseremoglou and Santos [1]	Condition-based fleet maintenance; DRL under partial observability	Health-state and prognostic uncertainty	Maintenance-timing decisions driven by health information; complementary to weekly execution-capacity uncertainty
Widmer et al. [4]	Aircraft line maintenance scheduling; simulation and RL	Short turnaround task and resource allocation	Local ground-stop scheduling, complementary to cross-week backlog propagation
Ruiz-Rodríguez et al. [37]	Dynamic maintenance; RL, GA simheuristic, and dispatching rules	Multi-method comparison under uncertainty	General dynamic maintenance comparison; the present study adds aircraft-specific proactive maintenance and rolling backlog
Zhang et al. [44]	Large-scale airline fleet maintenance; distributed hierarchical MARL	Multi-airport and fleet-scale coordination	Life-cycle multi-level coordination at a broader scale; the present study focuses on an executable weekly task pool service
Jia et al. [30]	Multi-objective aircraft recovery; memetic algorithm with RL	Evolutionary search and feasibility repair	Recovery and airport-slot objectives; complementary feasibility perspective to recurring maintenance-backlog dynamics
Liu et al. [38]	Hangar maintenance scheduling; chance constraints and hybrid PSO	Uncertain mission timing, multi-skilled resources, and rescheduling	Carrier wave-mission hangar scheduling; the present study addresses airline-style weekly rolling task pools
Present study	Weekly maintenance task pools; masked PPO with deterministic conversion	Backlog propagation, repair efficiency-dependent capacity, proactive maintenance, and paired independent instance evaluation	Executable weekly rolling architecture with explicit learned preference, feasibility, and physical execution layers

Table 1 highlights the integration pursued here: cross-week backlog propagation is coupled with repair efficiency-dependent executable capacity, a reproducible continuous-to-executable policy interface, and proactive maintenance within one rolling environment. The evaluation then separates long-horizon policy value, mechanism contributions, and robustness under common execution semantics.

3. Problem formulation

3.1. Weekly rolling maintenance scheduling model

Consider a fleet of N aircraft, indexed by i in $I = \{1, \dots, N\}$, and C maintenance task categories, indexed by c in $C = \{1, \dots, C\}$. The planning horizon contains T weekly decision epochs. At the beginning of each week, the effective task pool combines inherited backlog and newly arriving workload. The scheduler selects completed workload and may trigger an early maintenance/overhaul action. Table 2 summarizes the principal notation.

Table 2. Main notations used in the rolling maintenance model.

Symbol	Meaning
I	Number of aircraft
C	Number of maintenance task categories
T	Number of weekly decision epochs
$b_{i,c,t}$	Backlog of task category c for aircraft i at the start of week t
$a_{i,c,t}$	Newly arriving workload
$x_{i,c,t}$	Completed workload
$o_{i,t}$	Binary early maintenance/overhaul decision
$\delta_{i,c,t}$	Deterministic task reduction generated by the executed proactive maintenance transition
η_t	Weekly repair efficiency multiplier
R_t	Available weekly maintenance resource
$T_i^{\max}$	Weekly maintenance time limit for aircraft i
$A_{i,t}^{\text{avail}}$	Aircraft availability indicator
$q_{i,t}$	Policy task priority preference
$m_{i,t}$	Policy early-maintenance score

At the aggregate level, backlog evolves according to

$$B_{t+1} = B_t + A_t - X_t \quad (1)$$

At the aircraft-task level, let $\delta(i,c,t)$ denote the deterministic reduction in task-category c workload produced by the proactive maintenance state transition selected for aircraft i in week t :

$$b_{i,c,t+1} = \max\{0, b_{i,c,t} + a_{i,c,t} - x_{i,c,t} - \delta_{i,c,t}\} \quad (2)$$

Equation (2) makes the weekly process path-dependent: unfinished work enters the next decision state, while proactive maintenance can reduce specific future task pressure. The term $\delta(i,c,t)$ is generated directly by the legal maintenance state transition executed in AnyLogic for aircraft i and task

category c ; it equals zero when no proactive action occurs. This formulation directly ties backlog relief to executable maintenance state semantics rather than to a separately tuned coverage coefficient.

Figure 1 summarizes the weekly rolling process and separates four roles that are central to the proposed architecture: latent policy generation, policy-side feasibility handling, common physical execution, and evaluation. PPO produces latent preferences, deterministic projection converts them into an executable schedule, and the common executor applies the same physical and maintenance constraints to every method. Training reward signals guide policy learning, while cost, SVR, and CT remain external evaluation metrics.

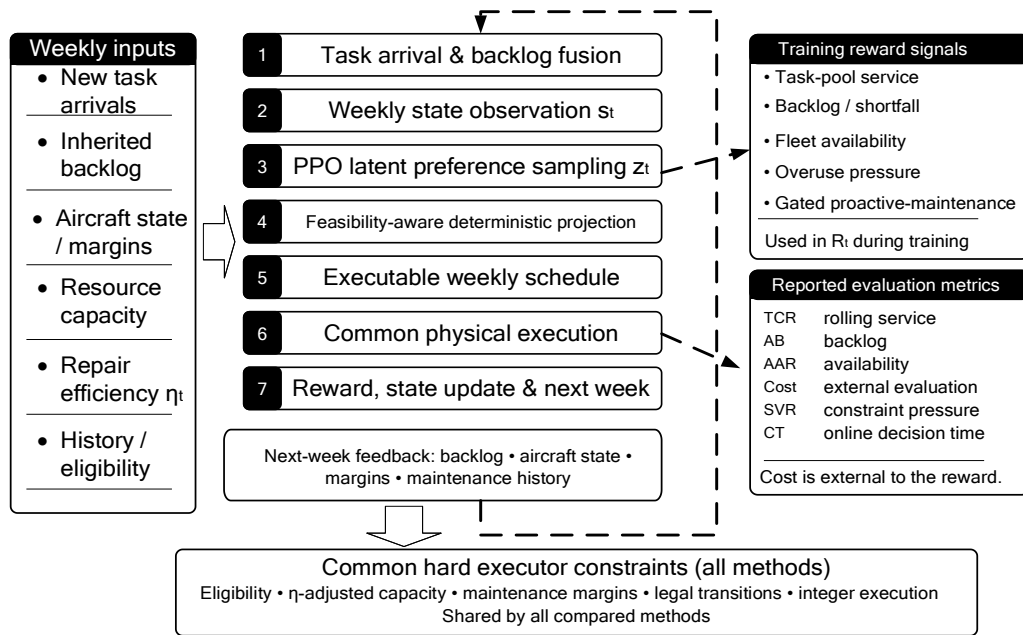


Figure 1. Weekly rolling scheduling process with separated feasibility, reward, and evaluation roles.

3.2. Repair efficiency uncertainty and observability assumption

The nominal processing time of task category c is denoted by the corresponding standard-time parameter. A weekly repair efficiency multiplier $\eta(t)$ converts standard workload into effective execution time:

$$p_{i,c,t}^{\text{eff}} = \bar{p}_c \eta_t \quad (3)$$

A value $\eta(t) > 1$ represents slower-than-nominal execution and reduces the standard workload that can be completed within a fixed weekly capacity; $\eta(t) < 1$ represents faster-than-nominal execution. The multiplier is generated from a truncated normal distribution:

$$\eta_t \sim \text{TruncNormal}(1, \sigma^2; 0.70, 1.50) \quad (4)$$

The study adopts explicit contemporaneous observability. In the full-state setting, $\eta(t)$ is sampled at the beginning of week t , provided to the policy, and used by the execution model to determine that week's effective capacity; future weekly values remain outside the current information set. This formulation isolates the scheduling value of current repair efficiency information under perfect contemporaneous observation. In the information-ablation policy, the policy-visible feature is fixed at the nominal value 1.0, while the physical executor continues to use the realized $\eta(t)$. Measurement and forecasting uncertainty constitute a separate extension of the present model.

The uncertainty-sensitivity experiments use $\sigma=0.05, 0.10, 0.20$ for the low-, medium-, and high-disturbance settings, respectively.

3.3. Aircraft state evolution and resource constraints

Let $H(i,t)$ be the accumulated flight-hour pressure, $h(i,t)$ the additional flight hours in week t , and $\Delta H_maint(i,t)$ the pressure released by maintenance. The state update is

$$H_{i,t+1} = H_{i,t} + h_{i,t} - \Delta H_{i,t}^{maint} \quad (5)$$

$$\Delta H_{i,t}^{maint} = \sum_{c \in \mathcal{C}} \rho_c x_{i,c,t} + \rho_{\text{overhaul}} o_{i,t} \quad (6)$$

The weekly resource constraint is

$$\sum_{i \in \mathcal{I}} \sum_{c \in \mathcal{C}} r_c x_{i,c,t} + \sum_{i \in \mathcal{I}} r_i^{\text{overhaul}} o_{i,t} \leq R_t \quad (7)$$

The eta-adjusted maintenance time constraint for each aircraft is

$$\sum_{c \in \mathcal{C}} \bar{p}_c \eta_t x_{i,c,t} + P_{i,t}^{\text{overhaul}} o_{i,t} \leq T_i^{\max} \quad (8)$$

Fleet availability is computed from the aircraft-level availability indicators:

$$A_t^{\text{fleet}} = \frac{1}{N} \sum_{i=1}^N A_{i,t}^{\text{avail}} \quad (9)$$

3.4. Evaluation metrics and interpretation

The model evaluates service capability, backlog, availability, cost, constraint pressure, and online computation. Task completion rate is calculated over the effective weekly task pools:

$$\text{TCR} = \frac{\sum_{t=1}^T \sum_{i=1}^N \sum_{c=1}^C x_{i,c,t}}{\sum_{t=1}^T \sum_{i=1}^N \sum_{c=1}^C (a_{i,c,t} + b_{i,c,t})} \quad (10)$$

Because unfinished work re-enters later weekly task pools, TCR measures the rolling service of the effective task pool. The aggregate simulator does not retain persistent work-order identities, so TCR is interpreted as a weekly service metric rather than an arrival-cohort completion probability. The average backlog and aircraft availability rate are

$$AB = \frac{1}{T} \sum_{t=1}^T \sum_{i=1}^N \sum_{c=1}^C b_{i,c,t} \quad (11)$$

$$AAR = \frac{1}{T} \sum_{t=1}^T A_t^{\text{fleet}} \quad (12)$$

SVR retains the abbreviation used by the simulation outputs and measures the proportion of weeks in which the configured flight hour, maintenance time, or availability pressure condition is activated. Let the binary weekly pressure indicator equal 1 when such a condition is active in week t and zero otherwise:

$$SVR = \frac{1}{T} \sum_{t=1}^T I_t^{\text{press}} \quad (13)$$

Throughout the paper, SVR is interpreted strictly as a model-internal scheduling constraint pressure metric. Cost is the composite maintenance execution cost reported by the simulator, and CT is the average weekly online decision time.

3.5. Hard constraints and soft scheduling objectives

The framework separates common physical execution constraints, policy-side feasibility guidance, and soft reward objectives. Common executor constraints apply to every method, while the proposed PPO adds deterministic feasibility-aware shaping before execution so that learning operates within an explicit operational envelope. Task pool service, backlog, fleet availability, and overuse pressure enter the reward, whereas cost, SVR, and CT remain external evaluation metrics. Table 3 summarizes these roles, and Figure 1 provides the process-level view.

Table 3. Execution constraints, policy-side feasibility guidance, and soft objectives.

Type	Item	Implementation
Common hard executor	Operational state and task eligibility	Ineligible or non-operational assignments are zeroed/rejected for every method.
Common hard executor	Eta-adjusted weekly capacity	Per-aircraft standard workload is capped by $\text{floor}\left(\frac{20}{\text{clip}(\eta, 0.70, 1.50)}\right)$ and maintenance margins.
Common hard executor	Maintenance-state legality	Executor permits only valid statechart transitions and legal proactive maintenance types.
Common hard executor	Integer execution	Largest remainder integerization and nonnegative clipping produce an executable workload.
Policy-side feasibility	Mask/prior/share cap/early gate/no-regret shield	Used by proposed PPO to reduce proposal-level infeasibility; neutralized in the action-masking ablation.
Soft reward objective	Task pool service and backlog	Completion rewards and backlog/shortfall penalties.
Soft reward objective	Fleet availability and overuse pressure	Availability reward/penalty and overuse pressure penalty; maintenance cost is reported separately as an evaluation metric.

4. Reinforcement learning–based rolling optimization method

4.1. Constrained Markov decision process

The weekly scheduling problem is represented as a constrained Markov decision process

$$M=(S,A,P,R,\gamma) \quad (14)$$

where S is the state space, A is the latent policy-action space, P is the weekly transition function, R is the reward, and γ is the discount factor. The state transition is

$$s_{t+1}=f(s_t, a_t^{\text{exe}}, \xi_t) \quad (15)$$

where $\xi(t)$ contains exogenous task arrivals, the repair efficiency multiplier, and flight-hour variation. The environment receives the projected executable schedule generated from the latent policy output and the current feasibility state.

4.2. State representation and policy architecture

The observation contains five features for each of 20 aircraft and 5 fleet-level features. Aircraft-level features are operational status, maintenance state code, minimum positive hours to maintenance, minimum positive weeks to maintenance, and hours flown. Global features are current task-arrival demand, inherited backlog, current repair efficiency state, fleet availability, and week index. Thus,

$$d_s=d_g+Nd_u=5+20\times 5=105 \quad (16)$$

The actor uses two 160-unit LayerNorm-Tanh hidden layers and a $2N$ -dimensional output; the critic uses the same hidden widths and a scalar output. The actor mean combines a physical maintenance margin prior and a bounded learned residual. Because the residual head is initialized at zero, the initial policy mean follows the physical prior, and PPO progressively learns state-dependent corrections to that prior. Actor and critic use separate hidden bodies. For $N=20$, the complete model contains 93,362 trainable parameters.

The observation and action interfaces scale with fleet size as $5N+5$ and $2N$, respectively, while the 160-160 hidden widths are retained. Thus, $N=10, 20, 30$ use 55/20, 105/40, and 155/60 observation/action dimensions, with 74,122, 93,362, and 112,602 trainable parameters. The $N=10$ and $N=30$ policies are trained at their own dimensions, providing size-specific implementations of the same parameterized architecture.

4.3. Latent policy output and deterministic schedule conversion

The actor parameterizes a $2N$ -dimensional continuous latent action. During training, the latent action is sampled from the actor distribution; during deterministic fixed-policy evaluation, the latent action is set to the actor mean. For $N=20$, the first 20 components describe task-allocation preferences, and the remaining 20 components describe proactive maintenance preferences:

$$a_t^{\text{lat}} = [z_t^q, z_t^m] \in \mathbb{R}^{40} \quad (17)$$

$$q_t = \text{softmax}(z_t^q), m_t = \text{sigmoid}(z_t^m) \quad (18)$$

The task channel is transformed by softmax and passed through a deterministic stabilization path. Non-operational or maintenance-occupied aircraft receive zero priority, the learned priority is blended with a physical capacity/margin prior, and a per-aircraft share cap limits excessive concentration. The actor, therefore, learns aircraft-level shares of the aggregate eligible weekly workload pool. The environment converts the normalized aircraft weights into standard workload quotas, caps each quota by contemporaneous eta-adjusted capacity and maintenance margins, and applies largest-remainder integerization. Task-category bookkeeping remains deterministic in the state-transition model, and exact residual ties are resolved by stable aircraft ordering, yielding a reproducible executable allocation.

The proactive maintenance channel is transformed by a sigmoid and evaluated jointly with backlog, current workload, fleet availability, and hours/weeks-to-maintenance margins. A common deterministic gate applies global feasibility conditions, near-due eligibility, a no-regret capacity shield, and a value threshold. At most, one aircraft can be selected for proactive maintenance in a week, and the legal hour-based maintenance type with the smallest positive remaining margin is used. Figure 2 summarizes the two policy branches, their deterministic merge into an executable weekly schedule, and the separate PPO likelihood path used during training.

$$a_t^{\text{exe}} = \Psi(s_t, q_t, m_t, M_t) \quad (19)$$

where $M(t)$ denotes the state-dependent feasibility mask. The mapping Ψ provides the key interface between the learned aircraft-level policy and the executable weekly schedule: PPO produces continuous aircraft-level preferences, and Ψ converts them into task allocation quotas and legal proactive maintenance actions. Task category eligibility and maintenance-state transitions remain deterministic within AnyLogic, while the largest remainder correction with stable ordering resolves integer residuals reproducibly.

4.4. Policy-side feasibility projection and common executor safeguards

Figure 2 separates policy-side feasibility guidance from the common physical executor. On the policy side, non-operational aircraft are masked before priority normalization, task shares are shaped by the capacity/margin prior and concentration cap, and proactive actions pass through the unified gate. The projected schedule then enters common executor safeguards shared by every method: task and operational eligibility, realized eta-dependent capacity, maintenance margins, legal state transitions, nonnegative clipping, and integer execution. This structure reduces proposal-level infeasibility while preserving the probability semantics of the latent PPO action.

4.5. Reward design and disclosed coefficients

Let $c(t)$ be the weekly completion rate, $q(t)=1-c(t)$, $a(t)$ the available aircraft ratio, $u(t)$ the unfinished workload, $o(t)$ the overuse pressure term, and $e(t)$ the early-maintenance count. The implemented weekly reward is

$$R_t = 18 + 128(c_t - 0.70) + I(c_t \geq 0.90) 52 \frac{c_t - 0.90}{0.10} - I(c_t < 0.86) 520(0.86 - c_t)^2 - I(c_t < 0.70) 160(0.70 - c_t)^2 + 32(a_t - 0.30) - I(a_t < 0.30) 120(0.30 - a_t)^2 - 0.36u_t - 0.005u_t^2 - 95q_t^2 - 26o_t + R_t^{\text{early}}. \quad (20)$$

$$R_t^{\text{early}} = \begin{cases} 0, & e_t = 0, \\ 2.80e_t, & c_t \geq 0.92, a_t \geq 0.32, u_t \leq 45, \\ 0.80e_t, & c_t \geq 0.88, a_t \geq 0.30, u_t \leq 65, \\ -9.00e_t - 1.60e_t^2 - 4.00e_t I(a_t < 0.30), & \text{otherwise.} \end{cases} \quad (21)$$

During training, the final transition of each episode receives an additional terminal backlog term $-0.60u(t)$. All reward coefficients are fixed before comparative evaluation and remain unchanged across the compared policies. The reward combines completion, availability, backlog, overuse, and proactive maintenance terms; maintenance cost is reported separately as an external evaluation metric. Table 4 summarizes the implemented components and their operational roles.

Table 4. Reward components and their operational roles.

Component	Implemented coefficient or condition	Role
Completion	$128(c-0.70)$; high-completion bonus above 0.90	Promote weekly task pool service
Low completion	Quadratic penalties below 0.86 and 0.70	Prevent persistent under-service
Availability	$32(a-0.30)$; quadratic penalty below 0.30	Preserve fleet readiness
Backlog	$-0.36u-0.005u^2-95(1-c)^2$	Suppress cross-week accumulation
Overuse pressure	-26o	Discourage operation near/exceeding limits
Early maintenance	State-dependent bonus or quadratic penalty	Reward proactive action under supportive operating conditions

4.6. PPO update and checkpoint use

During training, the actor samples the latent action in Equation (22). During deterministic fixed-policy evaluation, the actor mean is used directly, and the policy parameters remain unchanged.

$$a_t^{\text{lat}} \sim \pi_\theta(\cdot | s_t) \quad (22)$$

The rollout buffer stores each sampled latent action together with its old log-probability, and the new log-probability is evaluated for that same latent action. PPO likelihood is therefore computed exclusively in the latent action space. Projected priorities, proactive maintenance scores, and the executable CSV schedule are downstream execution objects.

$$r_t(\theta) = \frac{\pi_{\theta}(a_t^{\text{lat}} | s_t)}{\pi_{\theta_{\text{old}}}(a_t^{\text{lat}} | s_t)} \quad (23)$$

$$L^{\text{CLIP}}(\theta) = \mathbb{E}_t [\min(r_t(\theta) \hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon) \hat{A}_t)] \quad (24)$$

Each PPO update uses one complete 50-transition episode and seven optimization epochs. An approximate KL divergence above 0.03 triggers rejection of the update before backpropagation and optimizer execution. The checkpoint selected by the training criterion is then fixed for comparative evaluation. Training curves serve as optimization diagnostics, while all method comparisons use fixed-policy 50-week trajectories.

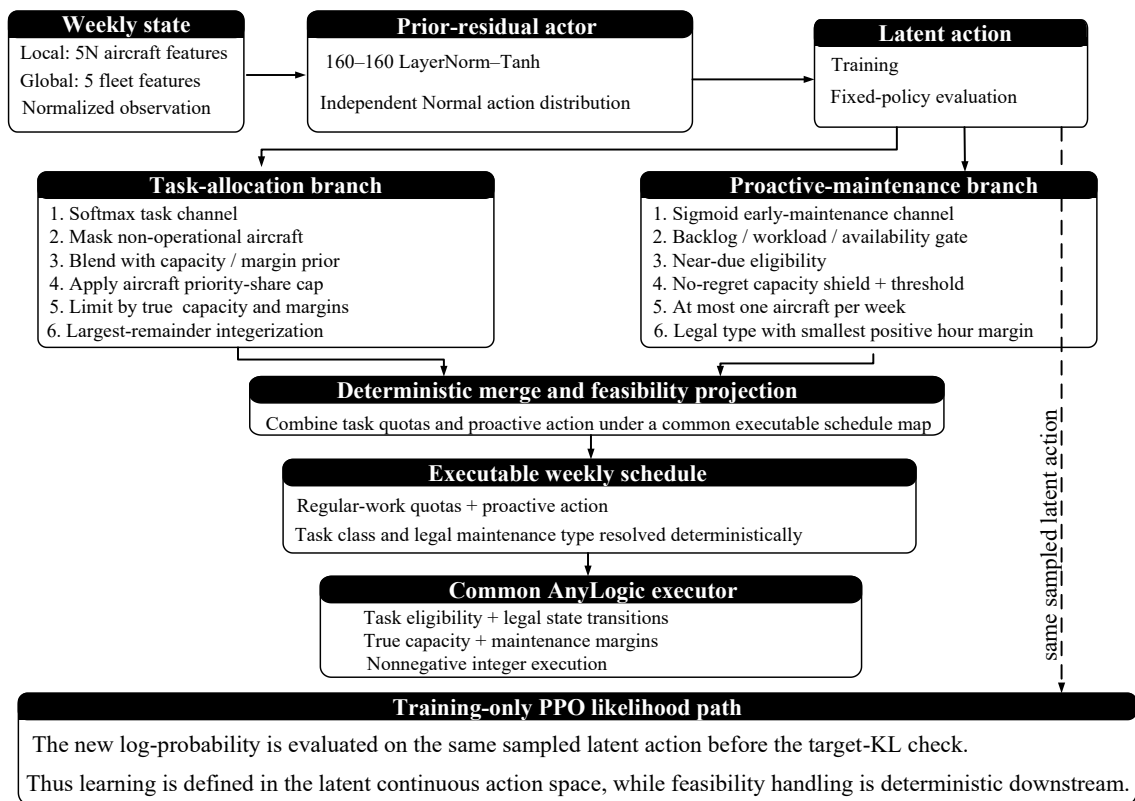


Figure 2. Latent-action policy, two-branch deterministic conversion, common execution, and PPO likelihood semantics.

4.7. Algorithm summary

Algorithm 1 summarizes policy training and fixed-policy evaluation; Algorithm 2 details the deterministic conversion from latent policy output to an executable weekly schedule. Together with Figure 2, the algorithms make the learned and deterministic parts of the scheduling pipeline explicit.

Algorithm 1. Training and fixed-policy evaluation procedure.

Step	Operation
1	Initialize the rolling environment, actor, critic, optimizer, and checkpoint records.
2	For each training episode, reset the 50-week task pool, fleet state, and resource state.
3	Observe the 105-dimensional weekly state; during training, sample the 40-dimensional latent action, whereas fixed-policy evaluation uses the actor mean.
4	Split the latent action into task and proactive channels, perform deterministic conversion, and apply policy-side feasibility projection.
5	Execute the weekly schedule in AnyLogic and observe the reward and the next state.
6	During training, store the sampled latent action and its matching old log-probability in the rollout buffer.
7	After the episode, compute GAE and perform PPO updates subject to the pre-step KL guard.
8	Save the current and best checkpoints; after training, freeze the selected checkpoint.
9	Run fixed-policy evaluation with paired task-arrival and repair-efficiency sequences with policy parameters fixed.

Algorithm 2. Deterministic priority-to-weekly schedule conversion.

Step	Operation
1	Split the latent action into 20 task allocation values and 20 early-maintenance values.
2	Apply softmax to the task channel and sigmoid to the proactive maintenance channel.
3	Mask non-operational or maintenance-occupied aircraft and construct the physical capacity/margin prior.
4	Blend learned task priorities with the physical prior; apply the first-week stabilization rule and per-aircraft priority share cap.
5	Allocate aircraft-level standard workload quotas proportionally under true η -adjusted capacity and maintenance margins; task category eligibility remains deterministic in AnyLogic.
6	Apply the largest remainder correction with stable aircraft ordering for exact ties to obtain executable integer workload.
7	Evaluate proactive candidates using backlog/load/availability, near-due eligibility, and legal maintenance-type margins.
8	Apply the no-regret capacity shield and value threshold; select at most one feasible aircraft and its tightest legal hours-based maintenance type.
9	Merge regular work quotas and proactive action into the executable weekly schedule and pass it to the common AnyLogic executor.

5. Experimental design*5.1. Synthetic simulation environment and scenario generation*

All reported experimental records are generated by the executable AnyLogic weekly maintenance environment. Each evaluation run contains 50 weekly decision epochs. At week t , the simulator supplies current demand, inherited backlog, aircraft and maintenance states, maintenance margins, the current repair efficiency multiplier $\eta(t)$, and the resulting executable capacity. The selected schedule

is executed before the next weekly state is formed, preserving a causal information sequence in which each decision uses the current state and current exogenous realization.

The base demand generator uses a Gaussian arrival multiplier, clamps the multiplier to $[0.70, 1.50]$, computes integer weekly demand from 180 times that multiplier, and then clamps demand to $[60, 300]$. The $N=20$ base evaluation replay contains 500 week-level demand values with an empirical multiplier mean of 0.9987 and a sample standard deviation of 0.0999; the mean weekly demand is 179.744. Within each comparison, the same exported demand sequence is replayed to every method for a given scenario, seed, and week. Demand-intensity robustness uses deterministic $0.8\times$ and $1.2\times$ multipliers applied to the same latent base demand before common integerization. Appendix A provides the complete executable generation and replay protocol used for these records.

Repair efficiency is sampled at the beginning of each week. For the main uncertainty study, η follows the truncated-normal model in Section 3 with support $[0.70, 1.50]$ and $\sigma=0.05, 0.10, 0.20$. The same realized η is used both in the environment capacity calculation and, for the full-state PPO only, as the contemporaneous policy feature. The no-efficiency-state policy observes a neutral value, while the real η remains active in the physical executor. For $N=20$, the nominal single-aircraft weekly standard work capacity is 20; the executable limit is $\text{floor}\left(\frac{20}{\text{clip}(\eta, 0.70, 1.50)}\right)$ and is further restricted by maintenance margins. At most, one proactive-maintenance event can be initiated per week.

5.2. Training, checkpoint selection, and independent-instance evaluation

The main $N=20$ full-state PPO policy is trained once for 500 episodes, with 50 weekly transitions per episode (25,000 training decisions). The checkpoint selected by the training criterion is fixed and evaluated over 10 paired seeds. The no-efficiency-state and no-action-mask variants are trained independently under the same 500×50 episode structure and evaluated from their own checkpoints. The proactive maintenance ablation reuses the full-state policy and neutralizes only the proactive channel during evaluation. The $N=10$ and $N=30$ fleet-size policies are trained independently at their corresponding observation and action dimensions, providing size-specific implementations of the parameterized architecture.

Each method-seed trajectory is evaluated in a fresh process/JVM/AnyLogic instance with matched demand and repair efficiency sequences and fixed policy parameters. Initialization, exogenous-input matching, and current information checks are verified before analysis. This independent instance design provides paired comparisons that are unaffected by execution order.

5.3. Compared methods and targeted component studies

The base comparison includes FCFS, Greedy-Urgency, exact weekly receding-horizon MILP, GA rolling heuristic, and the proposed masked PPO (hereafter, proposed PPO). FCFS serves the current task pool in arrival order, whereas Greedy-Urgency applies deterministic current-state urgency. The MILP provides an exact one-week benchmark with integer regular work variables and binary proactive decisions. It lexicographically maximizes current-week service J1, then urgency-weighted service J2, and finally minimizes resource/intervention cost J3. All 500 evaluation-week MILP decisions are certified exact optima of this predefined one-week objective, providing a strong current-week reference for the 50-week rolling comparison.

The GA uses the same current-week objective and a direct executable chromosome, with population 40, 50 generations, and 1902 fitness evaluations per week. On 100 independent calibration states, it matched the exact J1 and the complete lexicographic tuple in 100% of cases; across the 500 evaluation weeks, it matched exact J1 in 100% of weeks and the full tuple in 56.6%. Together, the MILP and GA create strong current-week optimization references against which the value of learned long-horizon state feedback can be assessed.

Three component studies quantify distinct mechanisms. The proactive maintenance study uses the same nominal $N=20$ model parameterization as the base comparison, but a separate paired replay set, and compares the proposed PPO with the same policy after neutralizing the proactive channel. The repair efficiency information study compares independently trained full-state and neutralized-eta-input PPO policies while retaining the realized eta in the physical environment. The action-masking study compares the proposed PPO with an independently trained policy that removes policy-side feasibility guidance while retaining the common physical executor. These designs separate intertemporal intervention, information value, and feasibility handling within the same rolling framework.

5.4. Implementation settings and PPO hyperparameters

Table 5 summarizes the simulation and policy interface settings, and Table 6 reports the PPO hyperparameters used during training.

Table 5. Simulation, observation, action, and implementation settings.

Parameter	Setting
Base fleet size	20 aircraft
Rolling horizon	50 weeks
PPO training	500 episodes $\times$ 50 weeks per independently trained policy
Fixed-policy evaluation	10 seeds $\times$ 50 weeks per comparison cell
$N=20$ observation	$105=20 \times 5 + 5$ (20 aircraft $\times$ 5 local features + 5 global features)
$N=20$ latent action	$40=20+20$ (20 task priority + 20 proactive scores)
Actor/critic hidden widths	160-160 LayerNorm-Tanh / 160-160 LayerNorm-Tanh
$N=20$ trainable parameters	93,362
Nominal per-aircraft weekly capacity	20 standard-work units
Executable capacity	$\text{floor}\left(\frac{20}{\text{clip}(\eta, 0.70, 1.50)}\right)$, further limited by maintenance margins
Base demand rule	Gaussian multiplier $\rightarrow$ clamp $[0.70, 1.50] \rightarrow \text{round}(180 \times \text{multiplier}) \rightarrow \text{clamp}[60, 300]$
Repair efficiency support	$[0.70, 1.50]$
Uncertainty intensities	$\sigma=0.05, 0.10, 0.20$
Proactive maintenance capacity	At most one aircraft per week
Environment/policy bridge	AnyLogic 8.9.9 / persistent PyTorch bridge
Hardware/software environment	Intel Core i9-14900K, ~ 63.72 GiB RAM, Windows 11, Java 17, CPU inference/training

Table 6. PPO training hyperparameters.

Hyperparameter	Value
Discount factor γ	0.985
GAE λ	0.94
PPO clip ϵ	0.16
Learning rate	2.0×10^{-4}
PPO epochs per 50-step rollout	7
Value-loss coefficient	0.65
Entropy coefficient	0.0035
Maximum gradient norm	0.50
Target KL	0.03; reject step before backward/optimizer when exceeded
Optimizer	Adam, $\epsilon = 1 \times 10^{-5}$
Initial action standard deviation	0.055
Mini-batch	None; one complete 50-transition episode per update

All PPO hyperparameters are fixed before comparative evaluation, and a single configuration is retained across the reported comparisons. The final configuration was established in pre-evaluation numerical stability tests covering finite policy outputs, actor-distribution scale, KL-guard behavior, and policy-environment bridge stability, and was then held fixed throughout the formal comparisons. The implemented weekly reward is fully specified in Equations (20)–(21) and Table 4, while maintenance cost is evaluated separately as an external performance metric.

5.5. Robustness, fleet-size evaluation, and statistical analysis

The evaluation spans three robustness dimensions. First, the fixed $N=20$ checkpoint is tested under exogenous-demand multipliers 0.8, 1.0, and 1.2. Second, the same checkpoint is tested under truncated-normal, uniform, and lognormal repair efficiency distributions with comparable location and dispersion; the uniform bounds are $[0.8267949, 1.1732051]$, and the lognormal distribution uses log-space $\mu=-0.0049752$ and $\sigma=0.0997513$ with rejection to $[0.70, 1.50]$. Third, fleet size varies across $N=10, 20, 30$ with approximately proportional demand scaling and observation/action dimensions $5N+5$ and $2N$. The $N=10$ and $N=30$ policies are trained at their own dimensions, so the fleet-size study evaluates size-specific behavior of the parameterized architecture.

Methods sharing a scenario and fleet size use identical seed-indexed exogenous demand and repair-efficiency sequences. Core outcomes are summarized as the arithmetic mean and sample standard deviation across $n=10$ seeds. When paired seed-level outputs are available, method differences are computed seed by seed and reported with two-sided 95% Student's t confidence intervals over the 10 paired differences ($df=9$); statistical resolution and operational effect size are interpreted separately. The demand intensity and distribution family studies follow the retained robustness outputs and are reported as 10-seed descriptive means. TCR denotes rolling task pool service, AB average backlog, AAR average aircraft availability, cost composite maintenance execution cost, SVR scheduling constraint pressure, and CT average weekly online decision time.

6. Results and discussion

6.1. Base-scenario multi-objective comparison

Under the independent instance, matched-input protocol of Section 5, Table 7 reports the base-scenario comparison across the five scheduling methods.

Table 7. Base-scenario comparison of five scheduling methods (mean $\pm$ sample SD, $n=10$).

Method	TCR/%	AB	AAR/%	Cost	SVR/%	CT/ms
FCFS	19.44 $\pm$ 0.65	609.86 $\pm$ 25.05	27.80 $\pm$ 0.67	9005.66 $\pm$ 64.78	90.40 $\pm$ 3.63	0.024 $\pm$ 0.003
Greedy-Urgency	14.78 $\pm$ 0.33	768.01 $\pm$ 17.46	44.01 $\pm$ 0.93	8636.89 $\pm$ 86.23	56.00 $\pm$ 2.11	0.032 $\pm$ 0.001
Exact weekly	13.83 $\pm$ 0.29	895.24 $\pm$ 20.62	14.88 $\pm$ 0.64	9773.66 $\pm$ 93.13	100.00 $\pm$ 0.00	243.018 $\pm$ 13.867
MILP						
GA rolling heuristic	15.21 $\pm$ 0.92	800.74 $\pm$ 35.24	24.31 $\pm$ 1.96	9439.06 $\pm$ 163.44	100.00 $\pm$ 0.00	168.961 $\pm$ 2.243
Proposed masked	16.83 $\pm$ 0.40	681.13 $\pm$ 18.87	38.89 $\pm$ 0.52	8747.06 $\pm$ 64.93	55.80 $\pm$ 2.20	5.204 $\pm$ 0.225
PPO						

Figure 3 complements Table 7 by displaying the five methods on the six reported operating dimensions with method-level 95% confidence intervals. The confidence intervals in the figure describe uncertainty in each method's mean; the seed-paired method differences are reported in the text below.

Table 7 and Figure 3 place the proposed PPO in a favorable multi-objective region. It achieves TCR 16.83%, AB 681.13, AAR 38.89%, cost 8747.06, and SVR 55.80%, combining stronger service and backlog control than Greedy-Urgency with substantially better availability, cost, and constraint-pressure performance than FCFS. The result highlights the value of a rolling policy that coordinates immediate workload service with longer-horizon fleet-state quality.

The paired comparisons quantify these gains. Relative to Greedy-Urgency, the proposed PPO increases TCR by 2.05 percentage points (95% CI [1.92, 2.19]) and reduces AB by 86.89 ([83.21, 90.57] in magnitude). Relative to FCFS, it increases AAR by 11.09 percentage points ([10.60, 11.58]), reduces cost by 258.60 ([235.98, 281.21]), and lowers SVR by 34.60 percentage points ([31.09, 38.11]). These complementary gains place the proposed PPO at a balanced rolling operating point across the principal service, readiness, cost, and constraint-pressure dimensions; Table 7 reports the complete metric vector.

The exact one-week MILP and GA further demonstrate the value of long-horizon state feedback. Relative to the exact MILP, the proposed PPO increases TCR by 3.00 percentage points (paired 95% CI [2.76, 3.25]) and reduces AB by 214.12 ([204.51, 223.72]). Relative to GA, the proposed PPO improves TCR by 1.62 percentage points (paired 95% CI for PPO-minus-GA [0.84, 2.40]) and reduces AB by 119.62 ([89.76, 149.47]). Online decision time is approximately 5.20 ms for PPO, compared with 168.96 ms for GA and 243.02 ms for MILP. These results show that learning cumulative rolling

returns can provide stronger long-horizon service and backlog control than repeatedly optimizing the observable current week.

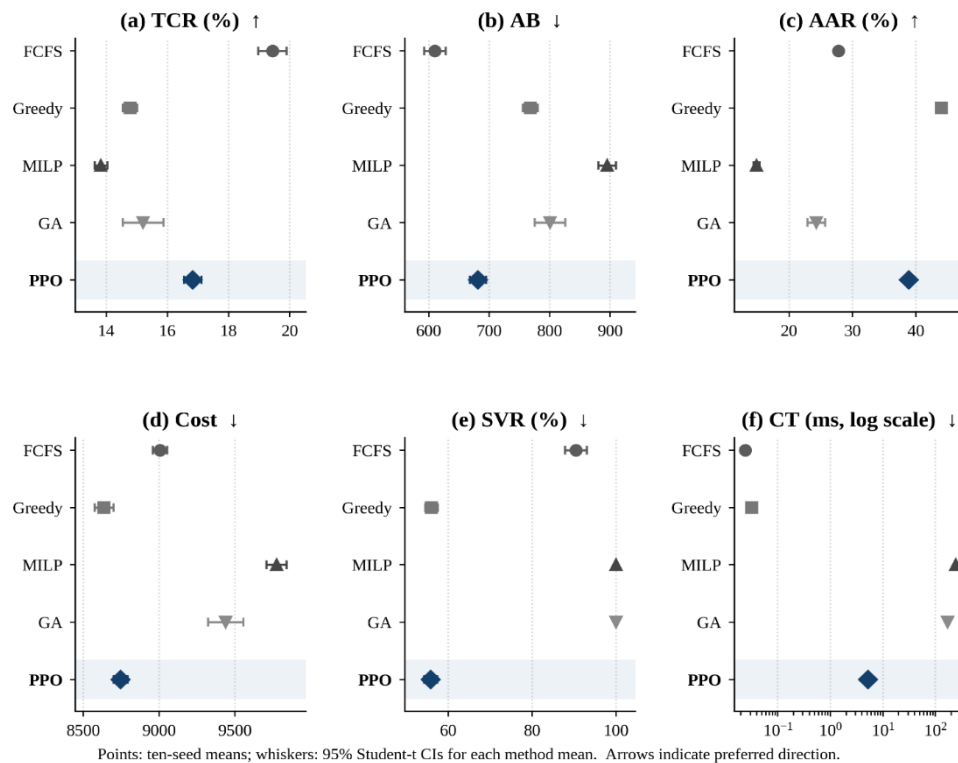


Figure 3. Base-scenario multi-objective operating points of the five scheduling methods (10-seed means with method-level 95% Student's t confidence intervals). The CT axis is logarithmic; arrows indicate preferred directions.

6.2. Sensitivity to repair-efficiency variability

Table 8 reports the sensitivity of the three principal scheduling methods to repair-efficiency variability.

Table 8. Sensitivity to repair-efficiency variability (mean $\pm$ sample SD, $n=10$).

Level	Method	TCR/%	AB	AAR/%	Cost	SVR/%
Low	FCFS	18.90 $\pm$ 0.96	631.14 $\pm$ 42.90	27.77 $\pm$ 0.69	9042.77 $\pm$ 120.87	89.00 $\pm$ 2.36
Low	Greedy-Urgency	14.58 $\pm$ 0.47	782.82 $\pm$ 35.47	43.60 $\pm$ 0.71	8694.14 $\pm$ 154.35	55.80 $\pm$ 2.39
Low	Proposed masked PPO	16.51 $\pm$ 0.57	697.92 $\pm$ 33.84	38.72 $\pm$ 0.80	8794.76 $\pm$ 145.66	56.60 $\pm$ 2.32
Medium	FCFS	19.13 $\pm$ 1.09	625.35 $\pm$ 45.96	27.58 $\pm$ 1.10	9061.02 $\pm$ 122.78	89.00 $\pm$ 2.36
Medium	Greedy-Urgency	14.67 $\pm$ 0.47	780.83 $\pm$ 35.48	43.63 $\pm$ 0.70	8718.48 $\pm$ 153.50	55.80 $\pm$ 1.99
Medium	Proposed masked PPO	16.60 $\pm$ 0.59	695.90 $\pm$ 34.00	38.77 $\pm$ 0.87	8812.79 $\pm$ 140.74	56.20 $\pm$ 1.99

Continued on next page

Level	Method	TCR/%	AB	AAR/%	Cost	SVR/%
High	FCFS	19.42 ± 1.11	618.67 ± 45.35	27.31 ± 1.24	9104.62 ± 122.74	89.60 ± 2.46
High	Greedy-Urgency	14.75 ± 0.48	779.05 ± 35.30	43.49 ± 0.78	8744.60 ± 150.44	56.40 ± 1.84
High	Proposed masked PPO	16.69 ± 0.65	694.74 ± 35.33	38.70 ± 0.87	8849.48 ± 138.18	57.20 ± 1.93

Table 8 provides the exact values, and Figure 4 shows the sensitivity profiles; seed-paired method differences are used for formal cross-method comparisons.

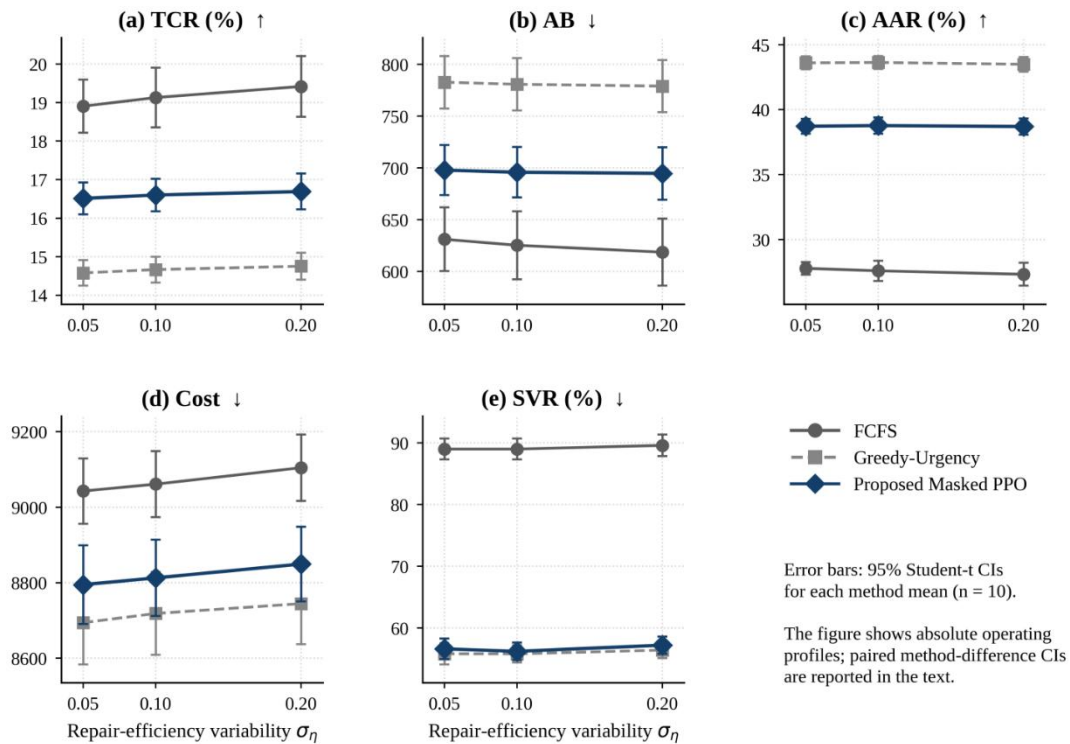


Figure 4. Sensitivity profiles for repair-efficiency disturbance standard deviation $\sigma = 0.05, 0.10$, and 0.20 . Points are 10-seed means, and error bars are two-sided 95% Student's t confidence intervals for each method mean. Higher TCR and AAR are preferable; lower AB, cost, and SVR are preferable. The plotted intervals characterize uncertainty in each method's mean; paired method-difference intervals are reported in the text.

Figure 4 shows that the principal advantages of the proposed PPO are retained across $\sigma = 0.05, 0.10$, and 0.20 . Relative to Greedy-Urgency, the proposed PPO increases TCR by 1.934, 1.935, and 1.938 percentage points, with paired 95% CIs of $[1.809, 2.058]$, $[1.806, 2.063]$, and $[1.778, 2.098]$, and reduces AB by 84.90, 84.93, and 84.31, respectively. Relative to FCFS, the proposed PPO increases AAR by 10.95–11.39 percentage points, reduces cost by 248.01–255.14, and lowers SVR by 32.4–32.8 percentage points. The repeated gains across all three variability levels demonstrate a stable multi-objective response to repair efficiency uncertainty.

Within the proposed PPO, the 10-seed means remain tightly grouped as variability changes: TCR stays within 16.51%–16.69%, AB within 694.74–697.92, AAR within 38.70%–38.77%, and SVR

within 56.20%–57.20%. These tight ranges indicate stable behavior over the tested repair efficiency variability range.

6.3. Intertemporal effect of proactive early maintenance

Table 9 reports the paired ablation of the proactive maintenance channel.

Table 9. Proactive early-maintenance ablation (mean $\pm$ sample SD, $n=10$).

Method	TCR/%	AB	AAR/%	Cost	SVR/%	CT/ms	Early count
Proposed masked PPO	16.63 $\pm$	691.62 $\pm$	38.40 $\pm$	8784.84 $\pm$	56.40 $\pm$	5.022 $\pm$	5.70 $\pm$
	0.59	28.76	0.64	81.90	1.58	0.299	0.67
PPO without early maintenance	16.17 $\pm$	705.08 $\pm$	39.61 $\pm$	8628.69 $\pm$	56.20 $\pm$	4.993 $\pm$	0.00 $\pm$
	0.64	30.80	0.43	78.43	1.14	0.193	0.00

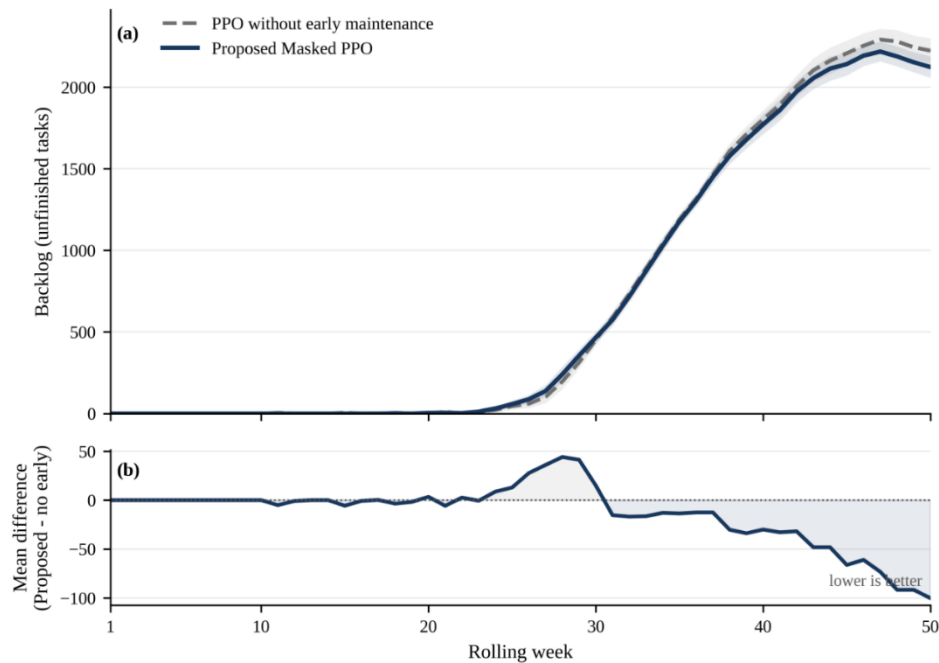


Figure 5. Fifty-week backlog trajectories with and without proactive early maintenance. (a) Lines are 10-seed means, and shaded bands are two-sided 95% Student's *t* confidence intervals for each method mean. (b) Descriptive difference between the two method means (proposed minus PPO without early maintenance); negative values indicate lower backlog under the proactive channel.

Table 9 quantifies the intertemporal contribution of proactive maintenance under the nominal $N=20$ setting using paired exogenous sequences. Activating the proactive channel raises TCR by 0.463 percentage points (paired 95% CI [0.338, 0.587]) and lowers AB by 13.46 (proposed-minus-no-early CI [-17.68, -9.24]). The proposed PPO initiates 5.70 ± 0.67 proactive maintenance events over 50

weeks, showing that a small number of targeted interventions can improve long-horizon service and backlog control.

Figure 5 shows how the backlog benefit accumulates over the rolling horizon. The two mean trajectories remain close during the early weeks; from week 31 onward, the proactive maintenance trajectory remains below the no-early trajectory, and the mean gap reaches 100.3 tasks by week 50. This delayed separation is consistent with the intended intertemporal role of proactive maintenance: selected current interventions reduce later maintenance pressure.

Together, Table 9 and Figure 5 identify proactive maintenance as a selective intertemporal capacity allocation mechanism: a small number of targeted interventions improves long-horizon task service and backlog control, while the complete operational response remains visible in Table 9.

6.4. Repair efficiency information sensitivity

Table 10 evaluates the marginal value of direct contemporaneous repair efficiency observation. An independently trained variant fixes the policy-visible eta feature at the neutral value of 1.0, while the physical executor continues to use the realized eta, and paired demand/eta sequences isolate the information-channel effect.

Table 10. Repair efficiency information sensitivity study (mean $\pm$ sample SD; paired difference = full PPO minus no policy-visible eta PPO, $n=10$).

Metric	Full PPO	No policy-visible eta	Full - no eta (95% paired CI)
TCR/%	16.74 $\pm$ 0.69	16.73 $\pm$ 0.68	+0.0069 pp [-0.0372, +0.0510]
AB	679.82 $\pm$ 29.99	679.53 $\pm$ 29.59	+0.284 [-1.442, +2.010]
AAR/%	38.89 $\pm$ 0.63	38.78 $\pm$ 0.66	+0.110 pp [-0.009, +0.229]
Cost	8689.57 $\pm$ 64.16	8683.46 $\pm$ 60.18	+6.11 [+0.15, +12.07]
SVR/%	56.40 $\pm$ 1.58	56.20 $\pm$ 1.75	+0.200 pp [-0.252, +0.652]
CT/ms	5.033 $\pm$ 0.181	5.109 $\pm$ 0.232	-0.0756 [-0.2626, +0.1115]

Neutralizing the direct eta observation leaves TCR, AB, AAR, SVR, and CT essentially unchanged; the corresponding paired intervals all include zero. Cost changes by only 6.11, about 0.07% of the 8683-unit mean of the neutralized input variant, and the mean proactive maintenance count remains 6.10 ± 0.57 events in both policies.

The information sensitivity result shows that the remaining state channels are sufficient for the tested policy to preserve its operating profile without relying on a dedicated eta observation channel. Together with Section 6.2, it separates the physical effect of repair efficiency from the marginal value of observing eta directly.

6.5. Action masking and feasibility control

Table 11 reports the paired action masking study using independently trained policies under the common executor.

Table 11. Action-masking ablation with independently trained policies (mean $\pm$ sample SD; paired difference = masked PPO minus no-mask PPO, $n=10$).

Metric	Masked PPO	No-mask PPO	Masked - no-mask (95% paired CI)
TCR/%	16.83 $\pm$ 0.40	17.75 $\pm$ 0.43	-0.918 pp [-1.099, -0.738]
AB	681.13 $\pm$ 18.87	658.04 $\pm$ 20.26	+23.09 [+17.02, +29.15]
AAR/%	38.89 $\pm$ 0.52	35.27 $\pm$ 0.90	+3.62 pp [+3.14, +4.10]
Cost	8747.06 $\pm$ 64.93	8982.02 $\pm$ 94.42	-234.96 [-268.46, -201.46]
SVR/%	55.80 $\pm$ 2.20	60.00 $\pm$ 3.77	-4.20 pp [-6.38, -2.02]
CT/ms	5.204 $\pm$ 0.225	4.841 $\pm$ 0.109	+0.362 [+0.158, +0.567]

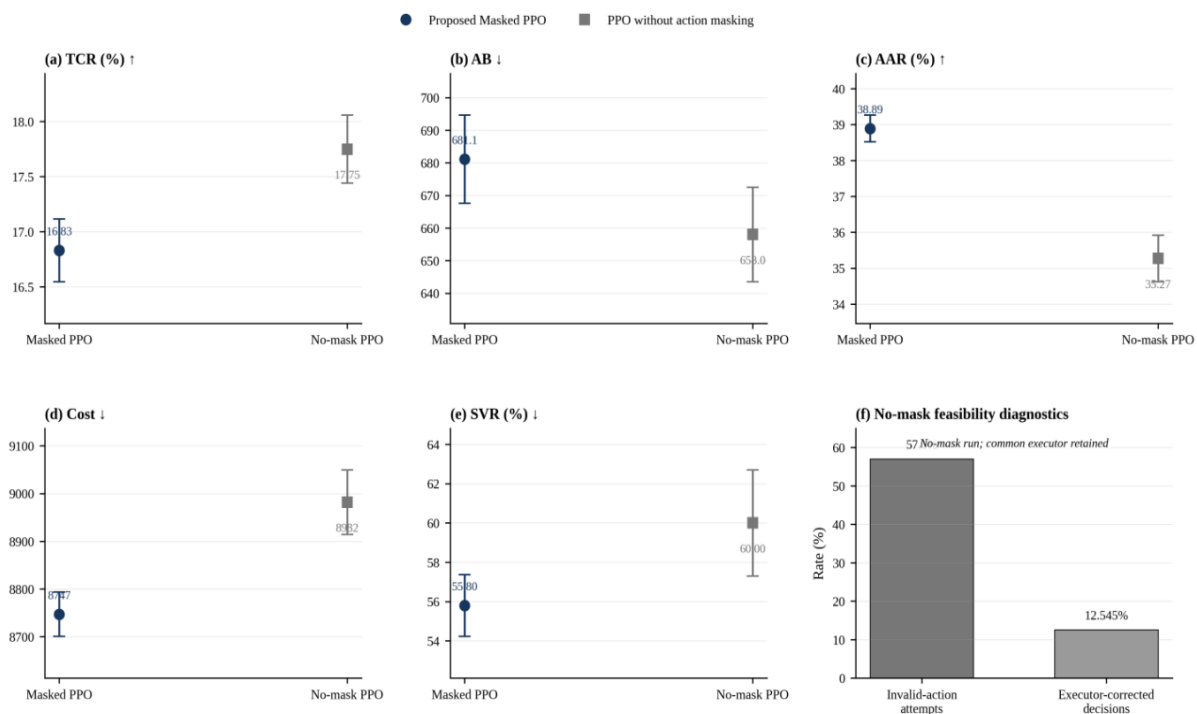


Figure 6. Action-masking study and no-mask feasibility diagnostics. Panels (a)–(e) show 10-seed method means with two-sided 95% Student's t confidence intervals. Panel (f) reports no-mask proposal diagnostics: 57.0% invalid action attempts and 12.545% decisions requiring common-executor correction.

Action masking strengthens the availability, cost, and feasibility dimensions of the learned policy. Relative to the no-mask policy, masking increases AAR by 3.62 percentage points (95% paired CI [+3.14, +4.10]), reduces cost by 234.96 ([-268.46, -201.46]), and lowers SVR by 4.20 percentage points ([-6.38, -2.02]). Service metrics move toward a more conservative allocation, as reported in the complete Table 11 metric vector. CT increases by only 0.362 ms ([+0.158, +0.567]), which is negligible on a weekly planning time scale. Figure 6 summarizes the performance gains together with the no-mask feasibility diagnostics.

The feasibility diagnostics show the practical value of policy-side guidance. Without the policy-side mechanisms, invalid action intentions appear in 57.0% of weekly decisions, and 12.545% of decisions require downstream correction by the common executor before execution. These diagnostics

quantify the proposal burden exposed when policy-side guidance is removed and motivate feasibility handling before the shared physical executor.

Action masking, therefore, serves as both a feasibility control and an operating-point mechanism. By screening and shaping proposals before the common executor, the architecture moves feasibility handling upstream and steers the policy toward a more favorable readiness-cost-constraint profile while retaining deterministic physical safeguards.

6.6. Robustness under demand and distribution shifts

6.6.1. Demand intensity robustness

Table 12 reports the demand-intensity OOD comparison under the fixed N=20 PPO checkpoint.

Table 12. Demand-intensity OOD robustness under the fixed N=20 PPO policy (10-seed descriptive means).

Demand level	Method	Mean demand	TCR/%	AB	AAR/%	Cost	SVR/%
Low	FCFS	143.804	38.48	210.31	40.72	7176.44	72.80
Low	Greedy-Urgency	143.804	24.90	334.57	57.02	6402.00	38.60
Low	Proposed masked PPO	143.804	33.86	233.20	51.03	6703.44	37.20
Nominal	FCFS	179.744	19.44	609.86	27.80	9005.66	90.40
Nominal	Greedy-Urgency	179.744	14.78	768.01	44.01	8636.89	56.00
Nominal	Proposed masked PPO	179.744	16.83	681.13	38.89	8747.06	55.80
High	FCFS	215.684	11.36	1206.62	19.26	10910.40	98.00
High	Greedy-Urgency	215.684	10.68	1283.63	34.36	10984.00	68.00
High	Proposed masked PPO	215.684	11.25	1220.47	30.66	10943.60	67.60

The demand-intensity test applies deterministic $0.8\times$ and $1.2\times$ multipliers to the same base-demand sequence used by the nominal demand case, producing realized mean weekly demands of 143.804, 179.744, and 215.684 for low, nominal, and high demand, respectively. The N=20 PPO checkpoint remains fixed across all three conditions. Figure 7 visualizes the resulting operating profiles. The retained robustness outputs provide 10-seed descriptive means, so this subsection focuses on effect magnitude and cross-method operating structure.

At $1.2\times$ demand, the absolute service, backlog, availability, cost, and constraint pressure metrics deteriorate relative to nominal demand, marking the upper end of the tested demand-capacity envelope; online CT remains near 5 ms. Table 12 retains the full response.

Across low, nominal, and high demand, the qualitative cross-method structure remains recognizable. Proposed PPO retains the lowest SVR at all three demand levels (37.20%, 55.80%, and 67.60%, respectively) and continues to occupy the intermediate region between the service-oriented FCFS and availability-oriented Greedy-Urgency policies. Figure 7 shows that the architecture preserves its characteristic multi-objective profile across the tested demand range.

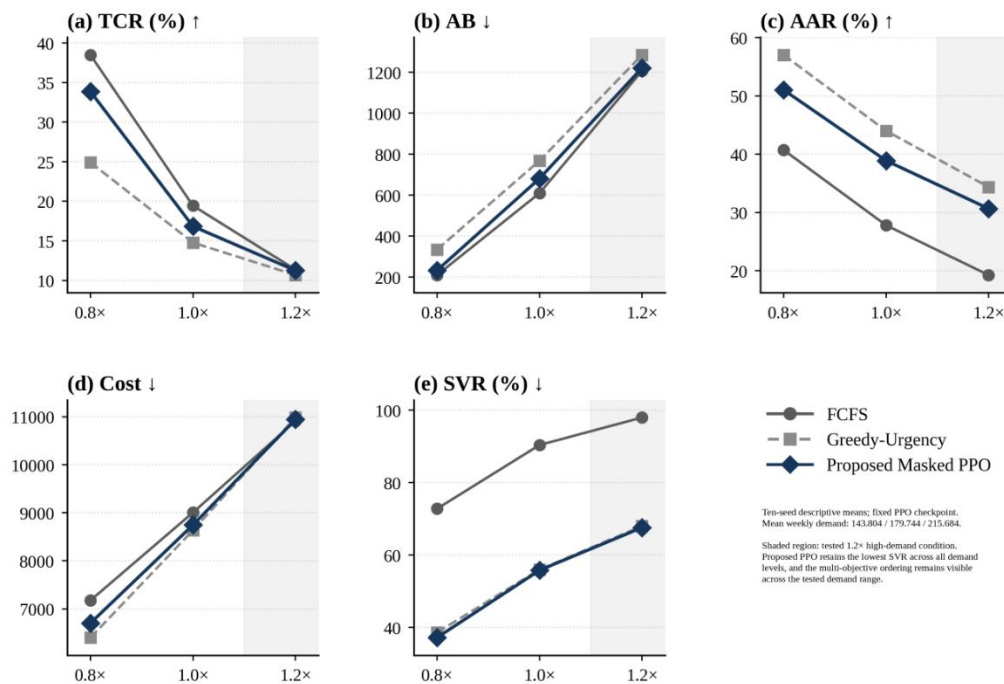


Figure 7. Demand intensity robustness under the fixed $N=20$ PPO policy. Lines show 10-seed descriptive means under $0.8\times$, $1.0\times$, and $1.2\times$ demand scaling; the realized mean weekly demands are 143.804, 179.744, and 215.684. The shaded region identifies the tested high-demand OOD condition.

6.6.2. Repair efficiency distribution-family robustness

Table 13 summarizes the distribution-family comparison under the same no-retraining protocol.

Table 13. Repair efficiency distribution-family robustness under the fixed $N=20$ PPO policy (10-seed descriptive means).

Repair-efficiency distribution	Method	Realized mean/SD	TCR/%	AB	AAR/%	Cost	SVR/%
Truncated normal	FCFS	0.9959/0.1057	19.44	609.86	27.80	9005.66	90.40
	Greedy-Urgency	0.9959/0.1057	14.78	768.01	44.01	8636.89	56.00
	Proposed masked PPO	0.9959/0.1057	16.83	681.13	38.89	8747.06	55.80
Uniform	FCFS	1.0107/0.1020	19.12	618.98	27.57	8994.04	90.60
	Greedy-Urgency	1.0107/0.1020	14.78	767.47	43.77	8635.46	56.60
	Proposed masked PPO	1.0107/0.1020	16.83	680.64	38.61	8744.94	56.60
Lognormal	FCFS	0.9869/0.0984	19.55	605.37	28.06	8988.60	89.80
	Greedy-Urgency	0.9869/0.0984	14.78	767.13	44.14	8631.23	56.00
	Proposed masked PPO	0.9869/0.0984	16.88	679.57	38.91	8762.13	55.20

The distribution-family study evaluates the fixed $N=20$ policy under a truncated normal, bounded uniform, and support-truncated lognormal repair efficiency model with comparable nominal location

and dispersion. Their realized 500-point eta mean/SD values are 0.9959/0.1057, 1.0107/0.1020, and 0.9869/0.0984, respectively. The close but nonidentical first and second moments define a practical distribution-family perturbation under comparable moments.

The operating outcomes vary little across the three tested families. For the proposed PPO, TCR ranges from 16.83% to 16.88% (maximum spread 0.053 percentage points), AB from 679.57 to 681.13 (spread 1.55), AAR from 38.61% to 38.91% (spread 0.30 percentage points), cost from 8744.94 to 8762.13 (spread 17.19), and SVR from 55.20% to 56.60% (spread 1.40 percentage points). Online CT remains between 4.869 and 5.204 ms. The principal cross-method ordering is likewise preserved: FCFS leads service and backlog, Greedy-Urgency leads availability and cost, and the proposed PPO remains between these two operating extremes, with SVR close between the proposed PPO and Greedy-Urgency.

Across the tested support and dispersion range, the repair efficiency distribution family has only a small descriptive effect on the operating point. The spreads in TCR, AB, AAR, cost, SVR, and CT are substantially smaller than those produced by demand scaling in Section 6.6.1, indicating limited sensitivity to the three bounded distribution families considered here.

6.7. Fleet size robustness and computational scaling

Table 14 summarizes the fleet size comparison using size-specific policies and approximately matched demand density.

Table 14. Fleet size robustness and online decision time under size-specific policies (10-seed descriptive means).

N	Method	TCR/%	AB/aircraft	AAR/%	Cost/aircraft	SVR/%	CT/ms
10	FCFS	21.42	26.96	24.78	447.91	88.00	0.020
	Greedy-Urgency	14.73	38.71	43.80	437.24	56.00	0.027
	Proposed masked PPO	17.13	33.82	37.48	446.43	55.60	3.463
20	FCFS	19.44	30.49	27.80	450.28	90.40	0.024
	Greedy-Urgency	14.78	38.40	44.01	431.84	56.00	0.032
	Proposed masked PPO	16.83	34.06	38.89	437.35	55.80	5.204
30	FCFS	18.48	32.23	29.58	449.78	91.00	0.031
	Greedy-Urgency	14.74	38.52	43.99	431.27	56.20	0.050
	Proposed masked PPO	16.76	34.23	39.09	436.29	56.00	7.173

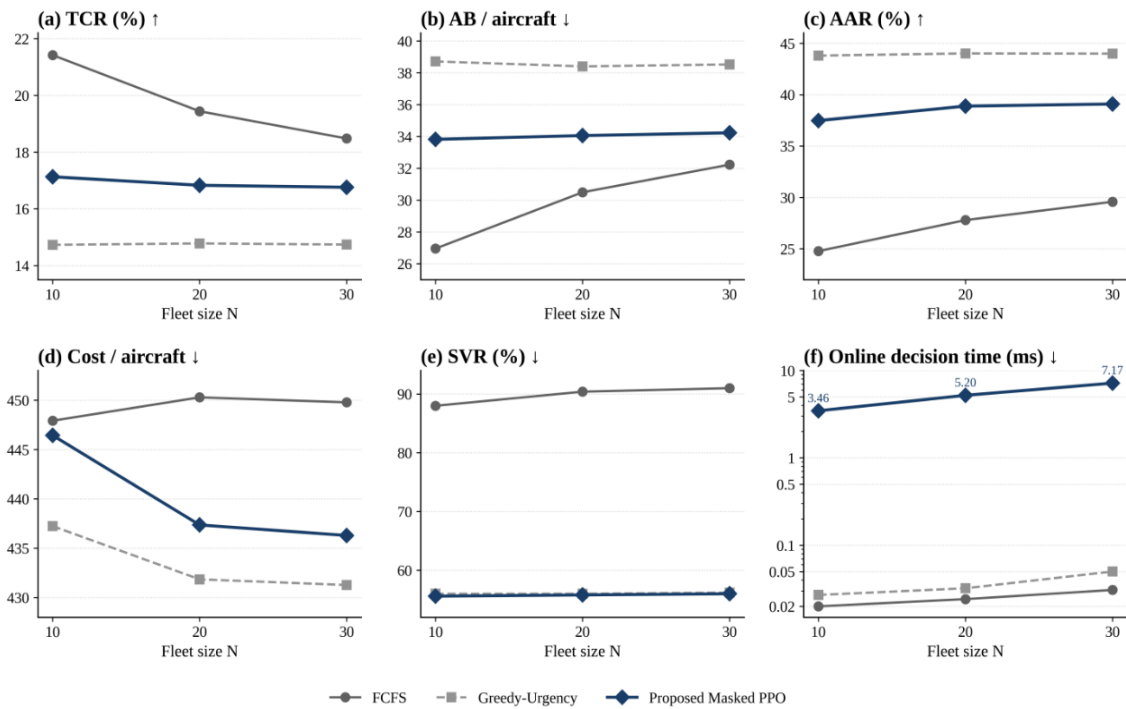


Figure 8. Fleet size robustness and online decision-time scaling for size-specific policies. Panels (a)-(e) show 10-seed descriptive means for the operational metrics. Panel (f) uses a logarithmic decision-time axis to display both rule-based and PPO values across $N=10$, 20 , and 30 .

Fleet size evaluation covers $N=10$, 20 , and 30 at approximately matched demand densities of 9.0126 , 8.9872 , and 8.9957 tasks per aircraft. The parameterized policy uses $5N+5$ observations and $2N$ latent actions, giving observation/action dimensions of $55/20$, $105/40$, and $155/60$ and trainable parameter counts of $74,122$, $93,362$, and $112,602$. $N=10$ and $N=30$ use independently trained policies at their corresponding dimensions. Table 14 and Figure 8 quantify size-specific behavior of the same parameterized architecture across the tested fleet range, with cross-fleet comparisons reported descriptively.

The proposed PPO remains highly consistent across the three tested fleet sizes. TCR changes only from 17.13% at $N=10$ to 16.76% at $N=30$ (-0.37 percentage points), and AB per aircraft changes from 33.82 to 34.23 ($+1.21\%$). Over the same range, AAR increases from 37.48% to 39.09% , cost per aircraft decreases from 446.43 to 436.29 , and SVR remains near 56% . The cross-method ordering is also stable, showing that the parameterized architecture preserves its characteristic operating profile as fleet size increases.

Online computation increases monotonically with fleet size but remains small relative to a weekly planning cycle. The proposed PPO decision time rises from 3.463 ms at $N=10$ to 5.204 ms at $N=20$ and 7.173 ms at $N=30$, while the network parameter count increases from $74,122$ to $112,602$. Recorded end-to-end training times are 979.84 s for $N=10$ and 5107.63 s for $N=30$ under the same Intel i9-14900K/Windows/PyTorch-CPU environment. These measurements provide an empirical scaling reference over $N=10$ – 30 .

6.8. Managerial implications and applicability

The experiments suggest three practical principles for weekly maintenance planning. First, decisions should account for cumulative fleet-state consequences: the proposed PPO combines stronger service and backlog control than Greedy-Urgency with markedly better availability, cost, and constraint pressure performance than FCFS, while the MILP and GA results show the limits of repeatedly optimizing a one-week objective. Second, proactive maintenance is most useful as a selective intertemporal intervention, and policy-side feasibility guidance moves feasibility handling upstream while improving readiness, cost, and constraint pressure before the common executor. Third, millisecond-level inference and stable behavior across repair-efficiency variation and $N=10-30$ support planner-in-the-loop use in which learned priorities remain transparent through deterministic schedule construction.

The present experiments use a synthetic aggregated weekly workload. Operational deployment would benefit from explicit work-order identities, personnel and spare-parts constraints, workstation interactions, and multi-base coordination, which define the next validation stage.

7. Conclusions

This study develops a feasibility-aware rolling decision framework for weekly aircraft maintenance in which dynamic task arrivals, inherited backlog, repair efficiency–dependent capacity, aircraft state constraints, and proactive maintenance are coordinated across repeated weekly transitions. The PPO policy learns aircraft-level latent preferences around a physical prior, while deterministic priority conversion, eligibility and capacity logic, integerization, and a common AnyLogic executor transform those preferences into executable schedules. The central methodological contribution is this transparent coupling of learned state feedback with explicit schedule construction and physical execution semantics.

The comparative results demonstrate strong complementary advantages across the principal operational objectives. Relative to Greedy-Urgency, the proposed PPO increases task pool service by 2.05 percentage points and reduces average backlog by 86.89. Relative to FCFS, it increases aircraft availability by 11.09 percentage points, reduces cost by 258.60, and lowers constraint pressure by 34.60 percentage points. The proposed PPO also achieves higher long-horizon TCR and lower backlog than both the exact one-week MILP and GA baselines, while requiring only about 5 ms per weekly decision. These results show that learning cumulative rolling consequences provides a practical advantage when current maintenance actions reshape future fleet states.

The mechanism studies show that proactive maintenance improves long-horizon service and backlog control, while action masking provides upstream feasibility control and strengthens availability, cost, and constraint pressure performance. The architecture also maintains a consistent operating profile across the tested repair efficiency intensities, bounded distribution families, and fleet sizes $N=10-30$, with online decision times of 3.463–7.173 ms. Together, these results establish a transparent and computationally efficient decision-support framework for path-dependent weekly aircraft maintenance scheduling under uncertain execution capacity.

Author contributions

Conceptualization, S.L. and Q.L.; Methodology, S.L. and P.C.; Software, P.C. and S.L.; Validation, S.L., M.Q. and P.C.; Formal analysis, S.L. and M.Q.; Investigation, S.L. and M.Q.; Data curation, P.C. and S.L.; Resources, Q.L.; Visualization, P.C. and S.L.; Writing - original draft, S.L.; Writing - review & editing, M.Q., P.C. and Q.L.; Supervision, Q.L.; Project administration, Q.L.; Funding acquisition, Q.L. All authors have read and approved the final version of the manuscript.

Use of Generative-AI tools declaration

Generative AI tools were used to assist language editing and manuscript organization. All scientific decisions, implementation choices, simulation results, and final interpretations were reviewed and approved by the authors, who take full responsibility for the final manuscript.

Acknowledgments

This research was supported by the Tsinghua University Initiative Scientific Research Program (20234616001), the 2023 Tsinghua University Undergraduate Education and Teaching Reform Project (69), the National Natural Science Foundation of China (No. 61771281), and the Science and Technology Innovation 2030 - Major Project of the New Generation Artificial Intelligence (2018AAA0101605).

Conflict of interest

The authors declare that they have no conflicts of interest.

Data and code availability

All reported data are synthetic outputs of the executable AnyLogic environment. The retained reproducibility package includes task-arrival and repair-efficiency sequences, week-level and seed-level evaluation records, solver certificates, trained checkpoints, input-matching records, and implementation files corresponding to the reported experiments. These materials are available from the authors upon reasonable request.

References

1. I. Tseremoglou, B. F. Santos, Condition-based maintenance scheduling of an aircraft fleet under partial observability: A deep reinforcement learning approach, *Reliab. Eng. Syst. Saf.*, **241** (2024), 109582. <https://doi.org/10.1016/j.res.2023.109582>
2. J. Lee, M. Mitici, Deep reinforcement learning for predictive aircraft maintenance using probabilistic remaining-useful-life prognostics, *Reliab. Eng. Syst. Saf.*, **230** (2023), 108908. <https://doi.org/10.1016/j.res.2022.108908>
3. P. Andrade, C. Silva, B. Ribeiro, B. F. Santos, Aircraft maintenance check scheduling using reinforcement learning, *Aerospace*, **8** (2021), 113. <https://doi.org/10.3390/aerospace8040113>

4. S. Widmer, S. Shaukat, C. L. Wu, Aircraft line maintenance scheduling using simulation and reinforcement learning, *Proc. 2023 Winter Simul. Conf. (WSC)*, 2023, 600–611. <https://doi.org/10.1109/WSC60868.2023.10408075>
5. I. Tseremoglou, P. J. V. Kessel, B. F. Santos, A comparative study of optimization models for condition-based maintenance scheduling of an aircraft fleet, *Aerospace*, **10** (2023), 120. <https://doi.org/10.3390/aerospace10020120>
6. J. P. Sprong, X. Jiang, H. Polinder, A deployment of prognostics to optimize aircraft maintenance: A literature review, *Annu. Conf. PHM Soc.*, **11** (2019). <https://doi.org/10.36001/phmconf.2019.v11i1.776>
7. I. Tseremoglou, M. T. Bieber, W. J. C. Verhagen, B. F. Santos, F. C. Freeman, P. J. V. Kessel, The impact of prognostic uncertainty on condition-based maintenance scheduling: An integrated approach, *AIAA Aviation 2022 Forum*, 2022. <https://doi.org/10.2514/6.2022-3967>
8. I. D. Pater, M. Mitici, Predictive maintenance for multi-component systems of repairables with remaining-useful-life prognostics and a limited stock of spare components, *Reliab. Eng. Syst. Saf.*, **214** (2021), 107761. <https://doi.org/10.1016/j.ress.2021.107761>
9. I. D. Pater, A. Reijns, M. Mitici, Alarm-based predictive maintenance scheduling for aircraft engines with imperfect remaining useful life prognostics, *Reliab. Eng. Syst. Saf.*, **221** (2022), 108341. <https://doi.org/10.1016/j.ress.2022.108341>
10. M. T. Bieber, W. J. C. Verhagen, B. F. Santos, An adaptive framework for remaining useful life predictions of aircraft systems, *PHM Soc. Eur. Conf.*, **6** (2021), 60–70. <https://doi.org/10.36001/phme.2021.v6i1.2868>
11. O. Ogunfowora, H. Najjaran, Reinforcement and deep reinforcement learning-based solutions for machine maintenance planning, scheduling policies, and optimization, *J. Manuf. Syst.*, **70** (2023), 244–263. <https://doi.org/10.1016/j.jmsy.2023.07.014>
12. R. Lamprecht, F. Wurst, M. F. Huber, Reinforcement learning based condition-oriented maintenance scheduling for flow line systems, *Proc. 2021 IEEE 19th Int. Conf. Ind. Inform. (INDIN)*, 2021. <https://doi.org/10.1109/INDIN45523.2021.9557373>
13. K. Vos, Z. Peng, E. Lee, W. Wang, Aircraft fleet availability optimisation: A reinforcement learning approach, *Aeronaut. J.*, **127** (2023), 2204–2218. <https://doi.org/10.1017/aer.2023.104>
14. V. Mattila, K. Virtanen, Scheduling fighter aircraft maintenance with reinforcement learning, *Proc. 2011 Winter Simul. Conf. (WSC)*, 2011. <https://doi.org/10.1109/WSC.2011.6147962>
15. R. S. Sutton, A. G. Barto, Reinforcement Learning: An Introduction, *IEEE Trans. Neural Netw. Learn. Syst.*, **9** (2018), 1054. <https://doi.org/10.1109/TNN.1998.712192>
16. T. A. Feo, J. F. Bard, Flight scheduling and maintenance base planning, *Manage. Sci.*, **35** (1989), 1415–1432. <https://doi.org/10.1287/mnsc.35.12.1415>
17. R. Gopalan, K. T. Talluri, The aircraft maintenance routing problem, *Oper. Res.*, **46** (1998), 260–271. <https://doi.org/10.1287/opre.46.2.260>
18. C. Barnhart, P. Belobaba, A. R. Odoni, Applications of operations research in the air transport industry, *Transp. Sci.*, **37** (2003), 368–391. <https://doi.org/10.1287/trsc.37.4.368.23276>
19. C. Barnhart, A. Cohn, Airline schedule planning: Accomplishments and opportunities, *Manuf. Serv. Oper. Manag.*, **6** (2004), 3–22. <https://doi.org/10.1287/msom.1030.0018>
20. B. Rexing, C. Barnhart, T. Kniker, A. Jarrah, N. Krishnamurthy, Airline fleet assignment with time windows, *Transp. Sci.*, **34** (2000), 1–20. <https://doi.org/10.1287/trsc.34.1.1.12277>

21. Q. Deng, B. F. Santos, R. Curran, A practical dynamic programming based methodology for aircraft maintenance check scheduling optimization, *Eur. J. Oper. Res.*, **281** (2020), 256–273. <https://doi.org/10.1016/j.ejor.2019.08.025>
22. C. Lagos, F. Delgado, M. Klapp, Dynamic optimization for airline maintenance operations, *Transp. Sci.*, **54** (2020), 998–1015. <https://doi.org/10.1287/trsc.2020.0984>
23. P. J. V. Kessel, F. C. Freeman, B. F. Santos, Airline maintenance task rescheduling in a disruptive environment, *Eur. J. Oper. Res.*, **308** (2023), 605–621. <https://doi.org/10.1016/j.ejor.2022.11.017>
24. Y. Qin, Z. X. Wang, F. T. S. Chan, S. H. Chung, T. Qu, A mathematical model and algorithms for the aircraft hangar maintenance scheduling problem, *Appl. Math. Model.*, **67** (2019), 491–509. <https://doi.org/10.1016/j.apm.2018.11.008>
25. D. T. Sanchez, B. Boyacı, K. G. Zografos, An optimisation framework for airline fleet maintenance scheduling with tail assignment considerations, *Transp. Res. Part B Methodol.*, **133** (2020), 142–164. <https://doi.org/10.1016/j.trb.2019.12.008>
26. H. Shahmoradi-Moghadam, N. Safaei, S. J. Sadjadi, Robust maintenance scheduling of aircraft fleet: A hybrid simulation-optimization approach, *IEEE Access*, **9** (2021), 17854–17865. <https://doi.org/10.1109/ACCESS.2021.3053714>
27. V. Mattila, K. Virtanen, Maintenance scheduling of a fleet of fighter aircraft through multi-objective simulation-optimization, *Simulation*, **90** (2014), 1023–1040. <https://doi.org/10.1177/0037549714540008>
28. Z. Liang, F. Xiao, X. Qian, L. Zhou, X. Jin, X. Lu, et al., A column generation-based heuristic for aircraft recovery problem with airport capacity constraints and maintenance flexibility, *Transp. Res. Part B Methodol.*, **113** (2018), 70–90. <https://doi.org/10.1016/j.trb.2018.05.007>
29. J. Lee, K. Lee, I. Moon, A reinforcement learning approach for multi-fleet aircraft recovery under airline disruption, *Appl. Soft Comput.*, **129** (2022), 109556. <https://doi.org/10.1016/j.asoc.2022.109556>
30. X. Jia, T. Guo, Y. Mei, M. Zhang, F. Xiao, X. Gu, et al., A self-adapting memetic algorithm with reinforcement learning for multi-objective aircraft recovery problems, *Appl. Soft Comput.*, **201** (2026), 115550. <https://doi.org/10.1016/j.asoc.2026.115550>
31. K. Ali, D. Rudinskas, V. Leonavičiūtė, Optimizing aircraft maintenance tasks allocation using mixed integer linear programming, *Aviation*, **29** (2025), 164–173. <https://doi.org/10.3846/aviation.2025.24535>
32. A. C. D. Castillo, A. K. Parlikad, Dynamic fleet management: Integrating predictive and preventive maintenance with operation workload balance to minimise cost, *Reliab. Eng. Syst. Saf.*, **249** (2024), 110243. <https://doi.org/10.1016/j.res.2024.110243>
33. S. Alaswad, Y. Xiang, A review on condition-based maintenance optimization models for stochastically deteriorating system, *Reliab. Eng. Syst. Saf.*, **157** (2017), 54–63. <https://doi.org/10.1016/j.res.2016.08.009>
34. K. Nguyen, P. Do, K. Huynh, C. Berenguer, A. Grall, Joint optimization of monitoring quality and replacement decisions in condition-based maintenance, *Reliab. Eng. Syst. Saf.*, **189** (2019), 177–195. <https://doi.org/10.1016/j.res.2019.04.034>
35. X. Liu, Q. Sun, Z. S. Ye, M. Yildirim, Optimal multi-type inspection policy for systems with imperfect online monitoring, *Reliab. Eng. Syst. Saf.*, **207** (2021), 107335. <https://doi.org/10.1016/j.res.2020.107335>

36. C. Song, C. Zhang, A. Shafieezadeh, R. Xiao, Value of information analysis in non-stationary stochastic decision environments: A reliability-assisted POMDP approach, *Reliab. Eng. Syst. Saf.*, **217** (2022), 108034. <https://doi.org/10.1016/j.ress.2021.108034>
37. M. L. Ruiz-Rodríguez, S. Kubler, J. Robert, Y. L. Traon, Dynamic maintenance scheduling approach under uncertainty: Comparison between reinforcement learning, genetic algorithm simheuristic, dispatching rules, *Expert Syst. Appl.*, **248** (2024), 123404. <https://doi.org/10.1016/j.eswa.2024.123404>
38. X. Liu, L. Wang, X. Li, M. Qian, X. Su, C. Lu, et al., Scheduling for carrier aircraft hangar maintenance in wave missions involving chance constraints, *Asia-Pac. J. Oper. Res.*, **43** (2026), 2540014. <https://doi.org/10.1142/S0217595925400147>
39. J. Chen, X. Zhou, Reinforcement learning based maintenance scheduling of flexible multi-machine manufacturing systems with varying interactive degradation, *Reliab. Eng. Syst. Saf.*, **260** (2025), 111018. <https://doi.org/10.1016/j.ress.2025.111018>
40. Y. Zhao, C. Smidts, Reinforcement learning for adaptive maintenance policy optimization under imperfect knowledge of the system degradation model and partial observability of system states, *Reliab. Eng. Syst. Saf.*, **224** (2022), 108541. <https://doi.org/10.1016/j.ress.2022.108541>
41. N. Zhang, W. Si, Deep reinforcement learning for condition-based maintenance planning of multi-component systems under dependent competing risks, *Reliab. Eng. Syst. Saf.*, **203** (2020), 107094. <https://doi.org/10.1016/j.ress.2020.107094>
42. K. G. Papakonstantinou, M. Shinozuka, Planning structural inspection and maintenance policies via dynamic programming and Markov processes. Part II: POMDP implementation, *Reliab. Eng. Syst. Saf.*, **130** (2014), 214–224. <https://doi.org/10.1016/j.ress.2014.04.006>
43. C. P. Andriotis, K. G. Papakonstantinou, Deep reinforcement learning driven inspection and maintenance planning under incomplete information and constraints, *Reliab. Eng. Syst. Saf.*, **212** (2021), 107551. <https://doi.org/10.1016/j.ress.2021.107551>
44. D. Zhang, Y. Hu, S. Zhang, Y. Zhang, Distributed hierarchical reinforcement learning for dynamic maintenance scheduling of large-scale airline fleets, *Reliab. Eng. Syst. Saf.*, **271** (2026), 112249. <https://doi.org/10.1016/j.ress.2026.112249>
45. C. Li, W. Han, H. Li, J. Liu, X. Wang, Y. Zhang, et al., Deep reinforcement learning for carrier-based aircraft flight deck operations scheduling problem, *Neural Netw.*, **200** (2026), 108776. <https://doi.org/10.1016/j.neunet.2026.108776>
46. X. Wang, X. Yu, Z. Wang, Z. Si, G. Wu, X. Su, et al., Machine learning-driven combinatorial optimization: A systematic review, *Arch. Comput. Methods Eng.*, 2026. <https://doi.org/10.1007/s11831-026-10679-4>



AIMS Press

© 2026 the Author(s), licensee AIMS Press. This is an open access article distributed under the terms of the Creative Commons Attribution License (<https://creativecommons.org/licenses/by/4.0>)