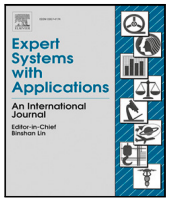
Expert Systems With Applications 248 (2024) 123404

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa



Dynamic maintenance scheduling approach under uncertainty: Comparison between reinforcement learning, genetic algorithm simheuristic, dispatching rules

Marcelo Luis Ruiz-Rodríguez [a,*], Sylvain Kubler [a], Jérémy Robert [b], Yves Le Traon [a]

[a] SnT, University of Luxembourg, 6 Rue Richard Coudenhove-Kalergi, L-1359 Luxembourg, Luxembourg
[b] Cebi Luxembourg S.A., 30 rue J.F. Kennedy, L-7327 Steinsel, Luxembourg

ARTICLE INFO

Keywords:
Maintenance
Manufacturing
Scheduling
Reinforcement learning
Metaheuristics

ABSTRACT

Maintenance planning and scheduling are an essential part of manufacturing companies to prevent machine breakdowns and increase machine uptime, along with production efficiency. One of the biggest challenges is to effectively address uncertainty (e.g., unexpected machine failures, variable time to repair). Multiple approaches have been used to solve the maintenance scheduling problem, including dispatching rules (DR), metaheuristics and simheuristics, or most recently reinforcement learning (RL). However, to the best of our knowledge, no study has ever studied to what extent these techniques are effective when faced with different levels of uncertainty. To overcome this gap in research, this paper presents an approach by analyzing the impact of categorized levels of uncertainty, specifically high and low, on the failure distribution and time to repair. Upon the formalization of the maintenance scheduling problem, the experiments conducted are performed in simulated scenarios with different degrees of uncertainty, and also considering a real-life manufacturing use case. The results indicate that rescheduling based on a genetic algorithm (GA) simheuristic outperforms RL and DR in terms of total machine uptime, but not in terms of the mean time to repair when configured with high re-optimization frequencies (i.e., hourly re-optimization), but rapidly underperforms when the re-optimization frequency decreases. Furthermore, our study demonstrates that GA-simheuristic is highly computationally demanding compared to RL and rule-based policies.

1. Introduction

In the manufacturing industry, production and maintenance processes are closely related. Although the objective of the production process is to meet the customer's demand for quality and production standards on time, it can be affected by constant wear and tear on machine components due to continuous use or environmental aspects, which results in machine breakdown and degradation of quality (Geurtsen et al., 2023). The maintenance process works to prevent and correct failures by restoring or replacing the components that caused the machine to fail.

Fig. 1 illustrates a possible processing flow to schedule maintenance tasks, starting from the collection of sensor data to estimate the remaining useful life (RUL) and/or assess whether the quality of manufactured products drops with time. This estimation/assessment can then be used to create/open maintenance tickets – *in addition to traditional preventive and corrective tickets* – that can finally be assigned to technicians optimally. The effectiveness of a maintenance schedule depends on the company's objectives (e.g., does the company want to minimize makespan, completion time, improve system availability or sustainability) and constraints (e.g., machine and technician priorities, dependencies between machines or processes, production schedules, *etc.*) (De Jonge & Scarf, 2020). Despite the prevalence of mathematical programming (Song et al., 2023; Stojanovic, 2023) and metaheuristic methods to solve maintenance scheduling problems, uncertainty remains a challenge (Guan et al., 2023; Zhuang et al., 2023) affecting the maintenance decisions mainly due to the assumption that parameters and their degree of uncertainty are known in advance, which usually not the case in practice (De Jonge et al., 2015). Uncertainty can come from different sources such as: estimation of RUL of equipment (Benaggoune et al., 2020), duration of maintenance time (Ying et al., 2016), availability and efficiency/fatigue of the technician (Ferjani et al., 2017), or the occurrence of unexpected events such as sudden failures and unavailability of the technician.

* Corresponding author.
E-mail addresses: marcelo.ruiz@uni.lu (M.L. Ruiz-Rodríguez), sylvain.kubler@uni.lu (S. Kubler), jeremy.robert@cebi.com (J. Robert), yves.letraon@uni.lu (Y. Le Traon).

https://doi.org/10.1016/j.eswa.2024.123404
Received 18 October 2023; Received in revised form 21 January 2024; Accepted 2 February 2024
Available online 8 February 2024

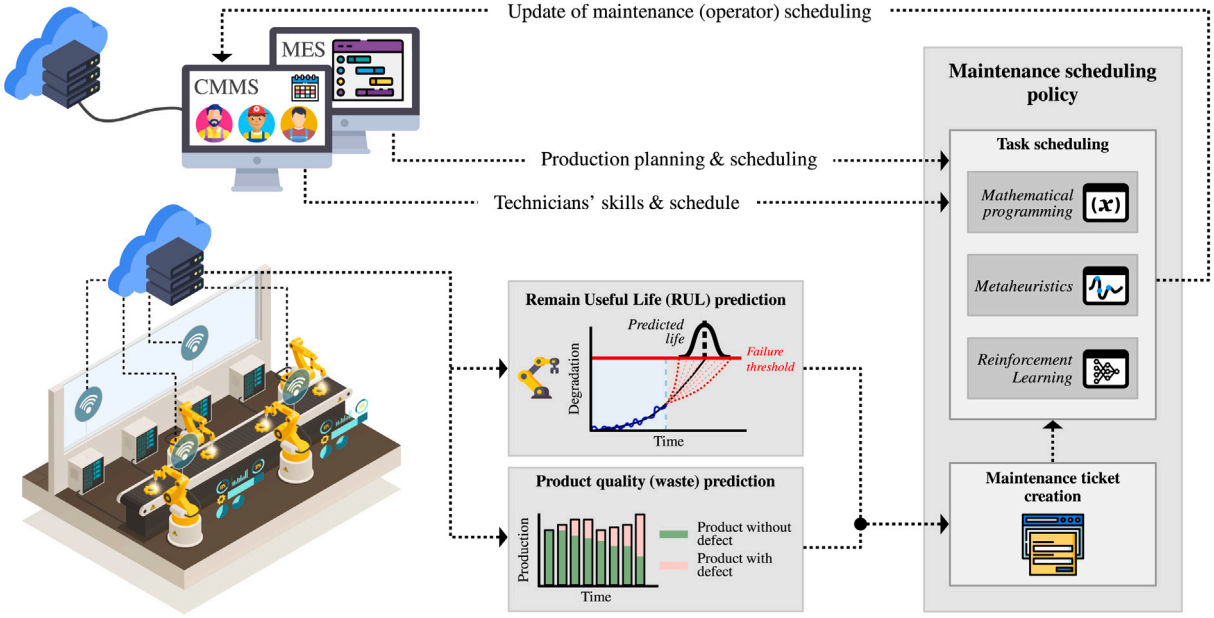


Fig. 1. Workflow diagram of the automated maintenance scheduling system.

To address this dynamic and stochastic problem, methods like hybrid simulation optimization (aka Simheuristic) (Alves & Ravetti, 2020), Machine Learning (Su et al., 2022), and Rule-based are often employed. One of these methods for hybrid simulation optimization is GA. This is a search and optimization algorithm inspired by the evolutionary principle, where solutions are obtained iteratively developed using genetic operators such as selection, crossover, and mutation. It is widely used to solve optimization problems such as exergoeconomic optimization for geothermal power plant (Nasruddin et al., 2018), stock market prediction (Deng et al., 2024), PID optimization (Zahir et al., 2020), communication networks (Neumann et al., 2023) among others. On the other hand, Reinforcement Learning (RL) is a type of machine learning in which an agent (decision maker) learns a policy through interaction with an environment with the objective of maximizing a reward. For the nature of RL methods to obtain an optimal (near-optimal) policy, several applications with different degrees of complexity and uncertainty have been implemented, such as vehicle routing (Pan & Liu, 2023), drone racing (Kaufmann et al., 2023), protein design (Lutz et al., 2023), and even sectors with high-risk implications like healthcare (Yang et al., 2023). Although these methods are widely used to optimize maintenance scheduling, there is still a lack of understanding of how they behave under different kinds of uncertainties. To fill this gap in the literature, this article aims to explore the performance of GA, RL, and classical DR, in terms of time to repair, machine uptime, and time complexity, under varying levels of uncertainty for machine failure distribution and maintenance duration.

Section 2 further discusses the maintenance scheduling problem, along with state-of-the-art methodologies and methods to solve it. Section 3 formally describes the maintenance scheduling problem and the optimization methods considered for benchmarking purposes (incl., RL, GA-Simheuristic, DR). Section 4 presents the results obtained considering both simulated maintenance scenarios and a real-life use case; discussion and conclusions follow. Note that all acronyms used in this paper are summarized in Table 1.

2. Scheduling under uncertainty

The possible origins of uncertainty in maintenance scheduling discussed in Section 2.1. Section 2.2 analyzes the current literature on static and dynamic maintenance scheduling, with a special emphasis on studies taking into account uncertainty. Based on this analysis, Section 2.3 discusses about the research gaps and how the present paper progresses the state-of-the-art.

2.1. Where does uncertainty come from?

The scheduling of maintenance tasks can be performed statically or dynamically. Static scheduling, which is performed offline, is adopted when the maintenance requirements are known in advance (e.g., list of tasks and technicians, job duration, technician's skill level, etc.), while dynamic maintenance is performed online to cope with the occurrence of unexpected events (e.g., sudden machine failures). The dynamic nature of the problem is highly linked to the uncertainty underpinning the maintenance process, whose most common types of uncertainty are (De Jonge & Scarf, 2020):

- **Maintenance type:** there is often uncertainty about what maintenance actions must be performed when a ticket is created such as component replacement, partial maintenance to restore a component's life, inspection, among others (Yu et al., 2019);
- **Maintenance duration:** the predicted duration of the maintenance can highly vary depending on the complexity of the operation and the capabilities/experience of the technician, or even depending on his fatigue at a given time (Ferjani et al., 2017);
- **Technician availability:** technician might unexpectedly request sick leave or vacation, or may need to be reassigned to a task of higher priority (Ruiz Rodríguez et al., 2022);
- **Machine availability:** unexpected failures may occur at any time, or a change of priority in the production schedule (Huang et al., 2020);
- **Failure distribution:** deterioration models of components are stochastic by nature, which may originate from natural conditions, such as natural wear and tear, or artificial due to human intervention (Sarazin et al., 2021);
- **Joint schedule:** when trying to take into consideration several schedules (e.g., maintenance and production schedules, aka. opportunistic maintenance), any abrupt change in one of the schedules (e.g., change in production frequency/priority) highly impacts the second schedule (Wocker et al., 2020);

To solve the maintenance scheduling problem, three main classes of methods can be used: *(i) Mathematical Programming:* set of techniques to

Table 1
Acronyms used in the present article.

Notation	Description	Notation	Description
AI	Artificial Intelligence	Ad	Artificial Data
At	Adjust	Av	Availability of system
CBM	Condition-Based Maintenance	CM	Corrective Maintenance
CMMS	Computerized Maintenance Management System	CNC	Computer Numerical Control machine
Cm	Corrective	Co	Completion Time
DBSCAN	Density-Based Spatial Clustering of Applications with Noise	DQN	Deep Q-Network
DR	dispatching rules	De	Deterministic
Dy	Dynamic	FIBT	First-In-Best-Technician
FIFO	First-In-First-Out	GA	Genetic Algorithm
In	Inspect	KPI	Key Performance Indicators
MDP	Markov Decision Process	MES	Manufacturing Execution System
ML	Machine Learning	MNNIA	Non-dominated neighbor immune algorithm
MP	Mathematical Programming	MTBF	Mean Time Between Failures
MTTR	Mean Time to Repair	Ma	Maintain
Me	Metaheuristics	Mk	Makespan
NSGA-II	Non-dominated Sorting Genetic Algorithm II	O	Others
PHM	Prognostic and Health Management	PM	Preventive Maintenance
PPO	Proximal Policy Optimization	PdM	Predictive Maintenance
Po	Production	Pr	Preventive
Pt	Profit	RL	Reinforcement Learning
RUL	Remaining Useful Life	Rc	Replace
Rd	Real Data	Re	Reliability of system
Rr	Repair	SB3	Stable-baseline-3
Sc	Service cost	Se	Stochastic
St	Static	Su	Sustainability
TTR	Time to repair	Ta	Maximum Tardiness
Uf	Failures	Um	Maintenance time
Up	Production	Ut	Tooling degradation
Ut	Technician availability	Uv	Machine availability

find the optimal solution (or feasible solutions in case of complexity or time constraints) but with the disadvantage of being computationally and time expensive; *(ii) Metaheuristics:* set of techniques to find near-optimal solutions but with a faster processing time than exact methods; and *(iii) Machine Learning (ML):* set of techniques that not only allow one to perform a search for the solution but also to train the algorithm to identify patterns to generalize for new instances of the problem.

2.2. Current state of affairs

Sections 2.2.1 and 2.2.2 provide an overview of research work dealing with static and dynamic maintenance scheduling, with a focus on approaches taken into account for uncertainty.

2.2.1. Static scheduling

The set of constraints and the complexity of the problem determine the type of scheduling to be used. Although some degrees of uncertainty are occasionally taken into account, static scheduling is often a deterministic problem, where all maintenance operations to be completed are known in advance. A large number of studies have used metaheuristics to solve such problems. Let us mention (Miao et al., 2022) who propose a modified non-dominated neighbor immune algorithm (MNNIA) for the allocation of technicians and spare parts, the objective being twofold: reducing both tardiness and maintenance cost. The authors compare MNNIA against other multiobjective algorithms under different conditions (i.e., varying the number of machines, technicians, and maintenance tasks), whose results show that MNNIA performs better in most cases. Yazdani et al. (2022) work on a joint production and maintenance problem aiming at reducing the total absolute deviation of completion time. The overall goal is to ensure that the machines have a similar completion time that is constrained by maintenance activities. To solve this problem, a Lion Optimization Algorithm is proposed, whose evaluation is carried out based on randomly generated cases considering different levels of complexity (varying the number of tasks from 50–600 and machines from 1–20). The algorithm is compared with six other metaheuristics, the results evidencing that it outperforms all of them. Other researchers such as Wocker et al. (2020) have used ML (clustering) techniques such as k-means, mean-shift, Expectation-Maximization, or Density-Based Spatial Clustering of Applications with Noise (aka DBSCAN) to solve static scheduling problems. The main problem was in the production of parts using unsupervised learning to apply opportunistic maintenance. Their approach was evaluated using real-life data from a car manufacturing plant based on a performance indicator showing *"how many stations can be stopped for maintenance while maintaining the desired production capacity?"*.

Unlike the above studies, some scholars have considered different types of uncertainty in the static scheduling problem. For example, Németh et al. (2020) propose a GA to select the most appropriate and cost-effective maintenance strategies and labor policies for production equipment considering the failures and availability of machines and technicians. The objective is to reduce the Life Cycle Cost, which is computed based on labor and production costs, maintenance costs, and on-demand service costs. Sun et al. (2019) worked with the problem of the saturation effect of maintenance actions, for which they proposed an algorithm to find the best maintenance intervals and the number of maintenance actions to decrease maintenance cost. The authors evaluate their approach using two types of mechanical systems, namely a high-precision boring mill and a power generation turbine system.

2.2.2. Dynamic scheduling

When there is a high level of uncertainty in the system, dynamic scheduling is considered and used more frequently. Until now, metaheuristics and iterative algorithms have been the preferred option (Geurtsen et al., 2023). In this respect, Mi et al. (2020) propose a multiobjective model using GA (NSGA-II) that aims to minimize maintenance costs and carbon emissions. The authors validate their approach on a cement production use case, whose results show that the reliability of the predictive maintenance plan is improved, while successfully

addressing the trade-off between carbon emission and maintenance cost. Ghaleb et al. (2020) also use GA to address the scheduling problem for a machine with multiple degradation states, the objective being to minimize the total system cost that consists of the inspection of the system, the machine repair, the completion time of the machine and the energy consumption costs. To evaluate the proposed mathematical model, different random instances were generated, and experiments show that the model outperforms other metaheuristics (incl., single-based metaheuristic and imperialist competitive algorithm). Celen and Djurdjanovic (2020) propose a decision-making model in a flexible manufacturing system with several stations that suffer from multiple degradations that cannot be perfectly observed; the goal being to maximize profits and decrease maintenance costs. The authors make use of the Tabu search algorithm, whose evaluation experiments consist of simulating a semiconductor manufacturing plant and comparing its algorithm with the traditional operation-independent and operation-dependent Condition-Based Maintenance (CBM) policies. The results show a significant improvement in the profit gain. Detti et al. (2019) work on the problem of joint scheduling of several jobs and maintenance activities considering uncertainty about maintenance duration, whose objective is to minimize makespan and total completion time. The authors implement different heuristics and make use of CPlex with randomly generated instances. Ruschel et al. (2020) address the problem of setting maintenance inspection intervals using process mining techniques and a probabilistic model in Bayesian networks. Their work is evaluated with the log of a lathe installed in a plant of the Brazilian automotive industry, whose results show an improvement in terms of cost and time. Lu et al. (2021) also use a Bayesian approach to predict future deterioration of product quality and machine reliability, thus optimizing maintenance costs, product quality, and machine reliability. Their model is applied to a boring mill used to manufacture a type of bearing seats and compared with a classical maintenance policy (considerable savings are obtained). Alves and Ravetti (2020) propose to use a Simheuristic approach to deal with uncertainty in the lot-sizing and scheduling problem in a maintenance system of identical parallel machines. The objective of maintenance scheduling is to define the periodicity of preventive maintenance to minimize the total time between preventive maintenance actions and corrective maintenance. Artificial data are used by varying different parameters exhaustively to show the results of the selection of different parameters for failure prevention. Arena et al. (2022) develop a framework called "maintenance driven scheduling cockpit" to support RUL-based maintenance and production scheduling decisions. The prototype is evaluated in different scenarios in a simulation establishing different costs for maintenance and production operations. Xia et al. (2021) proposes an energy-oriented joint optimization of the machine maintenance and tool replacement policy, in which energy consumption and cost are considered to decide the best maintenance action(s). The proposed policy is evaluated with respect to periodic preventive maintenance and energy-oriented preventive maintenance policies in a numerical example of a CNC machine, showing significant improvement.

Besides the use of metaheuristics and iterative algorithms, there has been an increasing attention to agent-based systems. Among other relevant studies, let us mention (Rokhforoz & Fink, 2021) who present an iterative distributed framework using a combination of model predictive control and Benders decomposition to address the problem of joint production and maintenance scheduling, where agents are used to model degradation of production units. The proposed framework is evaluated through the simulation of two distinct systems (food manufacturing and electrical production) and compared with centralized optimization and model predictive control methods. The results show that the proposed approach outperforms such methods, leading to significant computational power savings. Bencheikh et al. (2022) also developed a multiagent system for a joint production and maintenance scheduling policy considering Prognostic and Health Management (PHM) modules. Agents are used to subdivide the complexity of the system that consists of (i) an environment agent that controls the access to information to solve the subproblems; (ii) a supervisor agent that controls the access to the environment; (iii) a customer agent that corresponds to the manufacturing orders; (iv) a producer agent that manages each machine and its scheduling; (v) a maintenance agent that is responsible for providing maintenance tasks. One of the main insights of this work is the implementation of effective PHM modules to optimize the number of maintenance actions and machine availability. Kuhnle et al. (2019) propose an approach to determine the best window of opportunity to perform maintenance in a stochastic production environment. This approach uses multiple independent agents that learn the policy based on information from the production system buffer and time-to-failures. The RL policy outperforms Corrective Maintenance (CM) and Preventive Maintenance (PM) policies with respect to completed jobs, while evidencing how agents learn to execute maintenance closer to failure times with a low buffer volume. Valet et al. (2022) also tackle the problem of opportunistic maintenance using RL in a wafer manufacturing plant scenario. The proposed approach is compared with dispatching rules-based policies based on different performance indicators, including, among other indicators, order cycle time, machine downtime, and remaining lifetime. The work demonstrates the ability of a DQN agent to learn a joint competitive strategy for dispatching and opportunistic maintenance. Finally, let us mention (Yan et al., 2022) who addresses the problem of flexible job shop problem integrated with time-based maintenance and CBM. A RL algorithm called "double-layer Q-learning algorithm" is designed to select machines and jobs in a dynamic way. This work was evaluated against several metaheuristics, showing the relevance of the approach.

2.3. Research gaps & paper contribution

For ease of analysis and discussion of the remaining gaps in research, all research studies previously discussed have been summarized in Table 2 based on the following criteria:

- *Maintenance Policy:* Corrective (Cm), Preventive (Pr), and Predictive (PdM)
- *Maintenance action:* Inspect (In), Adjust (At), Maintain (Ma), Repair (Rr), Replace (Rc)
- *Scheduling type:* Static (St), Dynamic (Dy)
- *Environment:* Stochastic (Se), Deterministic (De)
- *Uncertainty:* Maintenance time (Um), Technician availability (Ut), Machine availability (Uv), Production (Up), Failures (Uf), Tooling degradation (Ut)
- *Objective:* Makespan (Mk), Maximum Tardiness (Ta), Completion Time (Co), Reliability of system (Re), Availability of system (Av), Profit (Pt), Service cost (Sc), Sustainability (Su), Production (Po)
- *Optimization methods:* Metaheuristics (Me), Machine Learning (ML), Mathematical Programming (MP), Others (O).
- *Evaluation:* Artificial Data (Ad), Real Data (Rd)

The literature analysis shows that most of the studies (≈57%) seek to minimize service costs as part of preventive and/or predictive maintenance policies, but very few seek to minimize maintenance time, while it can contribute to achieve substantial savings. Looking now at how and what type of uncertainty is tackled by the reviewed studies, it is interesting to note that most addresses the uncertainty of failure and very few addresses the uncertainty related to the duration of maintenance tasks (≈ 11%), while this duration can vary greatly from one technician to another. In the reviewed studies, Németh et al. (2020) consider uncertainty about the availability of technicians and Valet et al. (2022) and Detti et al. (2019) consider uncertainty about maintenance duration, but none of them consider/tackle both together. In the present paper, we propose three methods to jointly address these two types of uncertainty using three methods of different nature, namely RL, GA-simheuristic, and DR. In addition, our objective is to answer

Table 2
Summary table of the literature review.

Article	Policy	Actions	Sche.	Env.	Uncert.	Objective	Meth.	Eval.
Kuhnle et al. (2019)	PdM	Ma	Dy	Se	Uf	Po,Ri	ML	Ad
Detti et al. (2019)	Pr	N/A	Dy	Se	Um	Mk,Co	MP	Ad
Sun et al. (2019)	Pr	Ma,Rc	St	Se	Uf	Sc	O	Rd
Ruschel et al. (2020)	Pr	In	Dy	Se	Up	Av,Sc	ML	Rd
Alves and Ravetti (2020)	Cm,Pr,PdM	N/A	Dy	Se	Uf	Mk,Po,Ta	O	Ad
Mi et al. (2020)	PdM	Ma,Rr,Rc	Dy	Se	Uf	Sc,Su	Me	Rd
Celen and Djurdjanovic (2020)	Pr,PdM	Ma,Rc	Dy	Se	Ut	Pt,Sc	Me	Ad
Ghaleb et al. (2020)	PdM	In,Rr	Dy	Se	Uf,O	Pt,Sc	Me	Ad
Németh et al. (2020)	Cm,Pr,PdM	Rc	St	Se	Ut,Uv,Uf	Pt,Sc	Me	N/A
Wocker et al. (2020)	Pr	N/A	St	De	–	Po,Av	ML	Rd
Lu et al. (2021)	PdM	Rr,Rc	Dy	Se	Ut	Pt	N/A	Rd
Rokhforoz and Fink (2021)	PdM	Rc	Dy	Se	Ut	Po,Sc	MP	Rd
Xia et al. (2021)	Cm,Pr	Ma,Rc	Dy	Se	Uf	Re,Sc	N/A	Rd
Yazdani et al. (2022)	Pr	Ma	St	De	–	Co	Me	Ad
Miao et al. (2022)	Cm	Rc	St	De	–	Ta,Sc	Me	Ad
Valet et al. (2022)	Cm,Pr	Ma	Dy	Se	Um	Po,Av,Re	ML	Rd
Bencheikh et al. (2022)	PdM	N/A	Dy	Se	Uf	Po,Av,Re	O	Ad
Yan et al. (2022)	Cm,Pr	Rc	Dy	Se	Up,Uf	Mk	ML	Ad
Arena et al. (2022)	PdM	Rc	Dy	Se	Uf	Ta,Sc	O	Ad

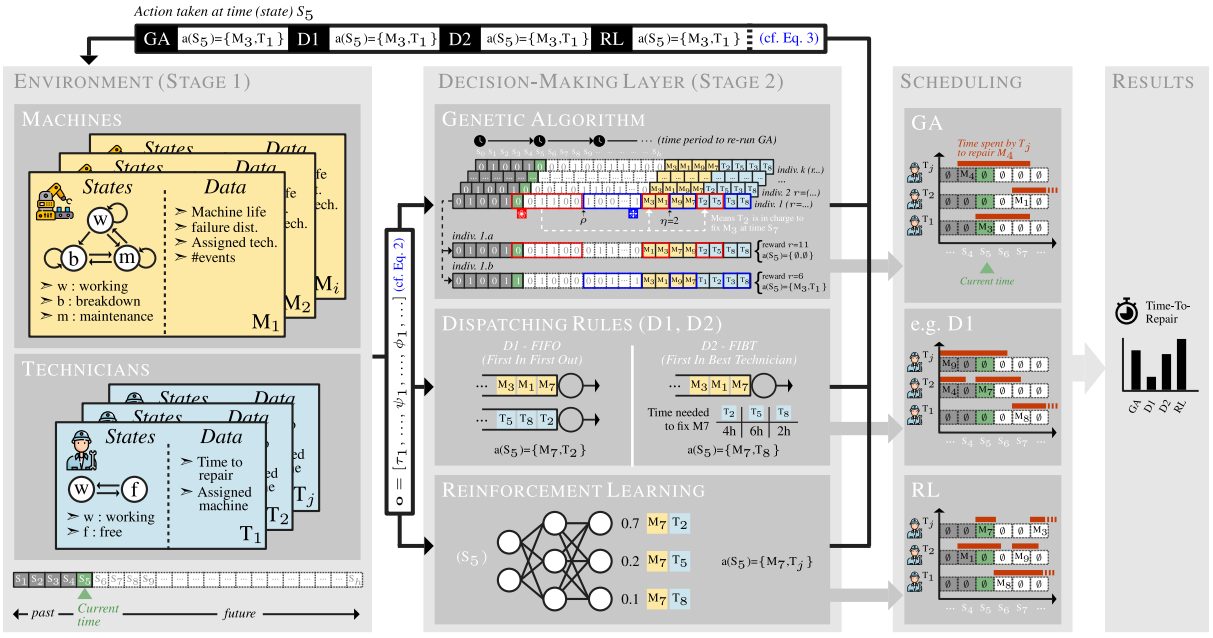


Fig. 2. Overview of the two-stage dynamic maintenance scheduling framework.

the research question of *"How does RL, GA-Simheuristic, DR perform when applied to different levels of uncertainty?*, which to the best of our knowledge has never been answered by the current literature. Section 3 presents the methodology to answer this research question.

3. Maintenance scheduling approaches

To evaluate and compare different optimization methods under different degrees of uncertainty, a two-stage approach is adopted, as illustrated in Fig. 2. The first step consists in building the environment to simulate the maintenance process in a job shop floor, where both machines and technicians are modeled (*cf.*, Fig. 2). The uncertainty associated with machines and technicians respectively relate to failure distribution and time to repair. The second stage consists in the definition of the three optimization methods (RL, GA-simheuristic, DR) and the way to compare them. These two stages are further detailed in Sections 3.1 and 3.2 respectively.

3.1. System description (environment)

The maintenance scheduling problem is defined as when and how to perform maintenance activities to maximize machine uptime. Let $\mathcal{M}$ be the set of machines. Each machine $i \in \mathcal{M}$ can fail at any time step t following a 2-parameter Weibull distribution $w_{it} \sim \mathcal{W}(\alpha_i, \beta_i)$ and changing its state, $x_{it} \in \{0, 1, 2\}$, from a working state ($x_{it} = 0$) to a breakdown state ($x_{it} = 1$). If machine $i \in \mathcal{M}$ fails, then a technician $j \in \mathcal{T}$ needs to perform maintenance on machine i and the technician's state changes from available ($y_{jt} = 0$) to busy ($y_{jt} = 1$), while the machine's state changes to maintenance ($x_{it} = 2$). Maintenance is assumed to be perfect, which means that machines are restored to the "As-Good-As-New" state. Each technician $j \in \mathcal{T}$ has different skills (e.g., one technician may be able to repair one type of failure in less time than another technician), so the intervention time for each machine $i \in \mathcal{M}$ is given by $d_{ij} \sim \lceil \mathcal{N}(\mu_{ij}, \sigma_{ij}) \rceil$, where μ_{ij} and σ_{ij} indicate the mean and standard deviation of the time to repair of technician j to machine i, respectively. A machine $i \in \mathcal{M}$ that is maintained by a technician $j \in \mathcal{T}$ in step t is represented by z_{ijt}. We assume that a single technician repairs a machine at a given step t, therefore $\sum_{j \in \mathcal{J}} z_{ijt} =$

Table 3
Nomenclature for variables used in the maintenance scheduling model.

Variable	Description
$\mathcal{M}$	The set of all machines
$\mathcal{T}$	The set of all technicians
D	Maintenance Log
$\mathcal{F}$	The set of all failures
H	Horizon of the scheduling
t	Timestep $0 \leq t \leq H$
α_i	Mean time of the failure for machine i
β_i	Standard deviation of the failure for machine i
x_{it}	State of the machine i at timestep t
y_{jt}	State of the technician j at timestep t
d_{ij}	Intervention time of technician i to machine j
η_{ijt}	Remaining maintenance time of technician i to machine j at time t
μ_{ij}	Mean of the time to repair for machine i by technician j
σ_{ij}	Standard deviation of the time to repair for machine i by technician j
z_{ijt}	Indicate if a technician j is doing maintenance to a machine i at the timestep t
s_k	Time of the ticket opening k from the maintenance log
c_k	Duration of the ticket k from the maintenance log
q_k	Indicates the technician who serviced the ticket k
f_k	Indicates the type of failure of the ticket k
τ_i	Normalized remaining time of the intervention of the ticket i
ψ_i	Normalized timespan since the last intervention of the ticket i
ϕ_j	Normalized remaining time of the technician j to be available

$\{0, 1\}$. The objective is to maximize the uptime of the machines in the complete horizon, which is equivalent to minimizing the MTTR while simultaneously reducing the breakdown time of the machines. The mathematical formulation can be written as in (1). Table 3 summarizes all the variables used in this paper.

$$\max \sum_{\forall t \in H} \sum_{\forall i \in \mathcal{M}} \mathbb{I}(x_{it} = 0) \tag{1}$$

Where $\mathbb{I}(x_{it} = 0)$ is an indicator variable that takes the value of 1 if the condition in the parenthesis is met, in this case, the machine i at the timestep t is in a working state. The goal of this objective function is to maximize the total number of time steps $t \in H$ where all machines $i \in \mathcal{M}$ are in a working state $x_{it} = 0$ across the entire time horizon H.

The problem of decision making in the context of the assignment of maintenance technicians can be formally represented as a Markov Decision Process (MDP), since we deal both with temporal dynamics (taking a maintenance decision at each time step t) and probabilistic uncertainty (about the potential failure of machines and the variable time to repair of technicians). MDP is a mathematical formulation of a problem in which an agent (decision maker) selects actions sequentially to transit through different states guided by rewards. An MDP can be expressed as a 5-tuple $\langle S, A, \mathcal{P}, R, \gamma \rangle$ consisting of (i) a state space S: indicating all possible states in which the agent can be in; (ii) an action space A: indicating all possible actions the agent can take; (iii) a transition function $\mathcal{P} : S \times A \rightarrow S$: indicating the probability of transitioning from any state $s \in S$ to state $s' \in S$ given that the agent took action $a \in A$; (iv) a reward function $R : S \times A \times S \rightarrow \mathbb{R}$: returning an immediate reward given by the transition from (s, a) to s'; and (v) a discount factor $\gamma \in [0, 1]$: indicating how myopic the agent is ($\gamma = 0$ indicates that the agent only cares about immediate reward, while $\gamma \rightarrow 1$ indicates that the agent gives more weight to future state information).

The state of the system, represented by different characteristics of the technician and the tickets, is represented by the vector $\mathbf{o}$. This vector provides three time-related features.

- **Remaining intervention time** (τ_i): For each ticket i, $\tau_i \in \mathbb{R}$ represents the normalized time left for the intervention to be completed.
- **Lifespan of the tickets** (ψ_i): Each ticket i, reflects the normalized duration or age since its creation represented by $\psi_i \in \mathbb{R}$
- **Technician availability** (ϕ_j): For each technician j, $\phi_j \in \mathbb{R}$ represents the remaining time of the technician to become available.

The state vector is represented as follows:

$$\mathbf{o} = [\tau_1, \ldots, \tau_{|\mathcal{M}|}, \psi_1, \ldots, \psi_{|\mathcal{M}|}, \phi_1, \ldots, \phi_{|\mathcal{T}|}] \tag{2}$$

In our system, the actions are defined as a selection process in which an available technician $t \in T$, is assigned to address and intervention $i \in \mathcal{M}$. The definition of the action space is given by:

$$\mathbf{a} \in \mathcal{M} \times \mathcal{T} \cup \{d\} \tag{3}$$

The Eq. (3) and its components can be understood as follow:

- $\mathbf{a}$: Is the action taken by the agent at a particular timestep
- $\mathcal{M} \times \mathcal{T}$: Is the Cartesian product of the set of interventions $\mathcal{M}$ and technicians $\mathcal{T}$. Represents all possible paring indicating which technician is assigned to which intervention.
- d : Indicated the scenario where no technician is assigned to any intervention.

The reward function is designed to provide a higher value as more machines are in a *working* state and decrease as the time remaining for maintenance increases, reflecting the cost of machines being out of service. The reward function, denoted as $r(\mathbf{o}, \mathbf{a}, \mathbf{o}')$, is defined as:

$$r(\mathbf{o}, \mathbf{a}, \mathbf{o}') = \frac{\sum_{i \in \mathcal{M}} (1 - \frac{\eta_{ijt}}{\max d_{ij}})}{|\mathcal{M}|} \tag{4}$$

Where each of the components of (4) are represented as:

- $r(\mathbf{o}, \mathbf{a}, \mathbf{o}')$: The reward function, dependent on the current observation $\mathbf{o}$, the action taken $\mathbf{a}$, and the next observation $\mathbf{o}'$.
- $(1 - \frac{\eta_{ijt}}{\max d_{ij}})$: The operational efficiency, η_{ijt} is the remaining maintenance time for machine i under technician j at time t, and $\max d_{ij}$ is the maximum possible maintenance duration for machine i and technician j. As this fraction represents the normalized remaining maintenance time, with the subtraction from 1 flipping the value to represent the efficiency (more remaining time means less efficiency).
- $\mathcal{M}$: Represents the total number of machines to normalize the reward.

In general, the agent learns a policy (π) to perform the best assignment since the reward is inversely proportional to the remaining time to repair; therefore, cases may arise where it is better to wait for a technician who will perform a fast intervention than to assign an available technician with a slow intervention time.

3.2. Decision-making layer

This layer defines the actions that should be performed based on the pending maintenance tasks and the information from the technician. Sections 3.2.1 to 3.2.2, respectively, introduce the proposed RL, GA-simheuristic, and DR to solve the dynamic maintenance scheduling problem formalized in Section 3.1.

3.2.1. Reinforcement learning

During exploration, the RL agent has to select a technician and machine to perform maintenance. One challenge consists in avoiding illegal actions such as *"the selected technician is already working on another machine"* or *"the selected machine is already under maintenance"*. There are usually two ways to solve this problem: (i) penalizing the illegal actions of the agent with a negative reward; (ii) using an action mask that allows the agent to select only actions that are valid and discard exploring non-viable solutions. In the present paper, we propose to use the Proximal Policy Optimization (PPO) (Schulman et al., 2017) with action masking based on the `Stable-baseline-3` (SB3) library implementation. PPO is part of the policy gradient methods and ensures

that the policy update stays close to the previous policy by optimizing a clipped surrogate objective. The idea of action masking is to apply a function directly on the raw logits of the actor's network to prevent the network from selecting actions as defined in 3.2.1–(6), where $l(s)$ is the logits generated based on state s and M is a large negative number. In this way, the policy can prevent the selection of actions due to the zero probability from the softmax function.

$$mask(l(s))_i = \begin{cases} l_i & \text{if } a_i \text{ is valid in } s \\ M & \text{Otherwise} \end{cases} \tag{5}$$

$$\pi'_\theta(\cdot|s) = \text{softmax}(mask(l(s))) \tag{6}$$

3.2.2. GA-simheuristic

GA is a popular alternative for the static scheduling problem (Németh et al., 2020), but it faces several challenges in dynamic scheduling such as: (i) representing all the states of an MDP in a chromosome may not be feasible, (ii) high execution time in relation to the number of states, or still (iii) managing uncertainty is not possible. To address these challenges, we adopt a strategy to obtain solutions in every n step with the current state of the system where we combine the genetic search for solutions using the MDP as a simulator. In our GA-simheuristic (GA-S), the encoding of the chromosome, at time t, is a vector of the form $[\mathbf{e}, \mathbf{b}, \mathbf{c}]$, where:

$$\mathbf{e} = [e_0, \dots, e_H] \qquad e_i \in \{0, 1\} \tag{7}$$

$$\mathbf{b} = [b_0, \dots, b_{|\mathcal{M}-1|}] \qquad b_i \in \mathcal{T} \tag{8}$$

$$\mathbf{c} = [c_0, \dots, c_{|\mathcal{M}-1|}] \qquad c_i \in \mathbb{Z} \tag{9}$$

The elements of $\mathbf{e}$ represent whether a maintenance action will be performed $e_i = 1$, or if no action, $e_i = 0$, will be performed for each time step t. The elements of $\mathbf{b}$ represent which technician will be assigned to perform the maintenance actions. The elements of $\mathbf{c}$ represent which machine will receive maintenance. This encoding allows one to preserve the sequential nature of actions. The vector $\mathbf{e}$ will generate two new children, $\mathbf{e^1}$ and $\mathbf{e^2}$, based on a point ρ, the elements $[e^1_0 : e^1_{\rho-1}]$ of the child $\mathbf{e^1}$ (as shown in Fig. 2 through ❀ symbol) will be permuted and the elements of $[e^2_\rho : e^2_H]$ of the child $\mathbf{e^2}$ (as shown in Fig. 2 through ❉ symbol) will be permuted. Let us consider $\eta = \sum_{i=0}^{\rho} e^1_i$, which is the number of maintenance operations that were performed up to ρ. Now, similar to the previous operation, the vector $\mathbf{b}$ and $\mathbf{c}$ will generate two new children each, $\mathbf{b^1}$, $\mathbf{b^2}$, $\mathbf{c^1}$, $\mathbf{c^2}$ based on η, the elements $[b^1_0 : b^1_{\eta-1}]$ and $[b^2_\eta : b^2_{|\mathcal{M}-1|}]$ will be mutated to select different technicians, while the elements $[c^1_0 : c^1_{\eta-1}]$ and $[c^2_{\eta-1} : c^2_{|\mathcal{M}-1|}]$ will be permuted to change the order of maintenance operations. The new children are then denoted by $[\mathbf{e^1}, \mathbf{b^1}, \mathbf{c^1}]$ and $[\mathbf{e^2}, \mathbf{b^2}, \mathbf{c^2}]$.

3.2.3. Dispatching rules

A simple way to perform maintenance scheduling is to establish rules or guidelines to determine how and in what order maintenance tasks should be performed. In this paper, we propose to use two rules for scheduling, namely *(i) First-In-First-Out (FIFO):* the first available task is assigned to the first available technician[1]; *(ii) First-In-Best-Technician (FIBT):* similar to FIFO except that the available technicians are evaluated regarding the maintenance task to be performed, the most suitable technician – *whose skills perform maintenance in the shortest time* – being assigned.

[1] In case the pairing cannot be performed, no maintenance action is performed.

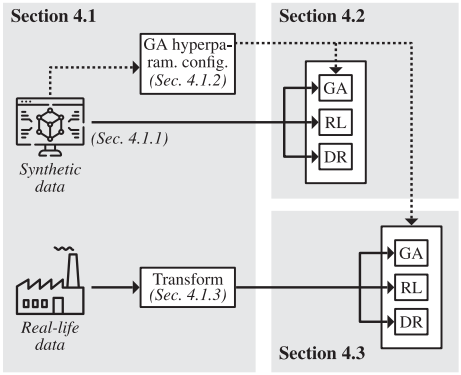


Fig. 3. Schematic representation of the organizational structure in Section 4.

4. Result and analysis

In this section, we present the evaluation of RL, GA-S, and DR to solve the dynamic scheduling problem previously defined (cf., Section 3.1), along with a financial discussion. To ease understanding of how this section is structured, we refer the reader to Fig. 3: Section 4.1 discusses the experimental settings, including how artificial/synthetic data is generated to consider different degrees of uncertainty (Section 4.1.1), how GA's hyperparameters are chosen (Section 4.1.2), and how real data from historical maintenance data from a manufacturing industry are processed/transformed to model the dynamic maintenance problem. Then, as highlighted in Fig. 3, both datasets (synthetic and real) are used as inputs of our experiments, whose results are presented and discussed in Section 4.2. Section 4.3 presents a case study on use of a manufacturing industry. Section 4.4 presents a brief discussion of the financial considerations to apply each policy in terms of the uptime of the machines.

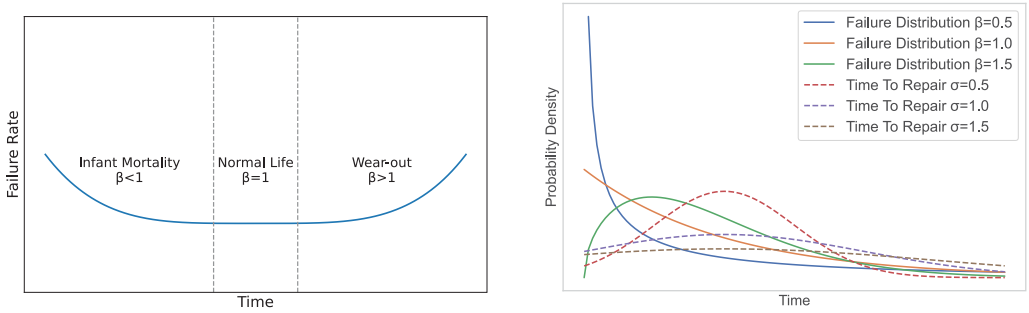
4.1. Experimental settings

4.1.1. Synthetic data

In reliability, the Weibull distribution is a continuous probability distribution commonly used to model the time to failure (Sulewski & Szymkowiak, 2022). It can be defined by 2-parameters, shape (β) and scale (α) parameters as shown in (10).

$$f(x; \alpha, \beta) = \frac{\beta}{\alpha}\left(\frac{x}{\alpha}\right)^{\beta-1} e^{-(x/\alpha)^\beta} \tag{10}$$

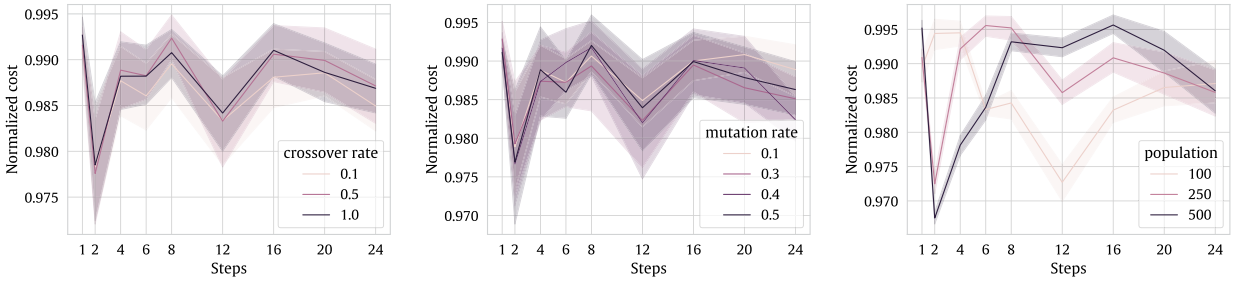
While the scale parameter represents a characteristic life, the shape parameter can characterize a specific type of failure. This type of failure can be observed in the bathub curve, where a $\beta < 1$ represents infant mortality, $\beta \approx 1$ a random failure, and $\beta > 1$ a wear-out effect (Jayatilleka & McLinn, 2021), as shown in Fig. 4(a). In our experiments, two degrees of uncertainty (low and high) for the failure distribution and time to repair are considered, which leads us to define four scenario configurations that combine all degrees of uncertainty. To vary the uncertainty of the failure distribution, $\beta \in \{1.2, 3.0\}$ is defined, which represents the levels of uncertainty. These values were selected to characterize the shape of the distribution where, 1.2 showed a skewed distribution while 3.0 tends to be a more symmetrical distribution. $\beta > 1$ establishes a wear-out model in the components where uncertainty decreases as $\beta \to \infty$, reducing the spread of the failures. Uncertainty in the time to repair is defined as $\sigma \in \{0.5, 1.5\}$, where uncertainty decreases as $\sigma \to 0$. These values were chosen to provide a stretched distribution, indicating lower degree of uncertainty in the distribution of failures (close to 0) and a longer characteristic of life being the components more resistant to failure, as shown in Fig. 4(b). Let us note that the proposed MDP has an horizon of 168 steps, equivalent to one week.



(a) Bathtub curve representing different failure rates over time

(b) Failure distribution and Time to repair with varying β, and σ parameters

Fig. 4. Illustration of failure rate and their corresponding phases in product life and model parameter uncertainty.



(a) Performance of crossover rates

(b) Performance of mutation rates

(c) Performance of population sizes

Fig. 5. Comparative analysis of genetic algorithm parameters: (a) Crossover Rates, (b) Mutation Rates, and (c) Population sizes, over 24 steps.

Table 4
Parameter Configuration for GA-S.

Variable	Value
Population Size	100, 250, 500
Mutation Rate	10%, 30%, 40%, 50%
Crossover Rate	10%, 50%, 100%
Stopping Criterion	5 min
Selection Strategy	Tournament

Table 5
Optimal genetic algorithm hyperparameters at each step interval.

	Number of steps								
	1	2	4	6	8	12	16	20	24
Cross.	1.0	1.0	0.5	0.5	0.5	1.0	1.0	0.5	0.5
Mut.	0.3	0.3	0.5	0.4	0.5	0.1	0.1	0.1	0.1
Pop.	500	100	100	250	250	500	500	500	100

4.1.2. GA-S hyperparameter optimization

We conducted a series of experiments with the objective of finding the best setting of the GA's parameters: population size, mutation rate, and crossover rate. To this end, a stopping time criteria (set to 5 min) is defined, along with a tournament selection strategy. Experiments are carried out 20 times with $\beta = 2.0$ and $\gamma = 1.0$. In total, 324 configurations are tested, which results from the combination of 3 Population size × 4 Mutation Rate × 3 Crossover Rate × 9-step configurations. In each of these configurations, the objective function values are normalized to ease comparison and identification of the best hyperparameter setting. Let us note that one of our goals is to select a parameter setting that would be flexible and robust to any operator's decision to modify the number of steps when performing dynamic scheduling. Table 4 summarizes the different hyperparameters selected.

In the GA-S approach, a defined re-optimization period must be defined to perform the rescheduling. In this respect, we propose to define $G \in \{1, 2, 4, 6, 8, 12, 16, 20, 24\}$ in which the search for the optimal solution will be performed. Although a timestep can be of different unit time (sec, min, hour), we selected those numbers in reference to a day (1 to 24), which means that one timestep refers to 1 h in our environment. Figs. 5(a), 5(b), 5(c) show the performance comparison between the different crossover rates, mutation rates and population size against the other parameters. The x-axis shows the different number of steps in which the re-optimization/rescheduling is performed, while the y-axis shows the values of the normalized objective function. Fig. 5(a) presents the comparative analysis of various crossover rates in all mutation rates and population sizes. The first configuration, which involves 1-step rescheduling, executes the scheduling process 168 times (as the MDP's horizon is of 168 steps). At the other extreme, the 24-step configuration carries out rescheduling only seven times, as it reaches the end of the horizon. The values obtained are very close to each other and it is difficult to conclude any trend with the crossover rate. Looking now at Fig. 5(b), which shows the comparative analysis of various mutation rates in all cases of crossover rates and population sizes, one may note a slight improvement associated with a mutation rate of 0.1 against the others for most of the steps. Finally, Fig. 5(c) shows a comparative analysis of population sizes in different time steps. The results suggest that a smaller population size is more advantageous in high-frequency rescheduling, characterized by low time-step values, as it enables increased crossover and mutation operations and enhances the exploration of diverse solutions. However, by decreasing the frequency of rescheduling, which leads to a higher complexity of the problem, a larger population size becomes more effective as it covers a wider range of solutions. As a summary, Table 5 gives insights into the best configuration results per re-optimization period (G).

4.1.3. Real data transformation

In this section, we conduct the experiments on a real-life use case from a manufacturing industry. We used 10-year historical maintenance data in which we pre-processed the data to extract the parameters for the maintenance duration and failure distribution.

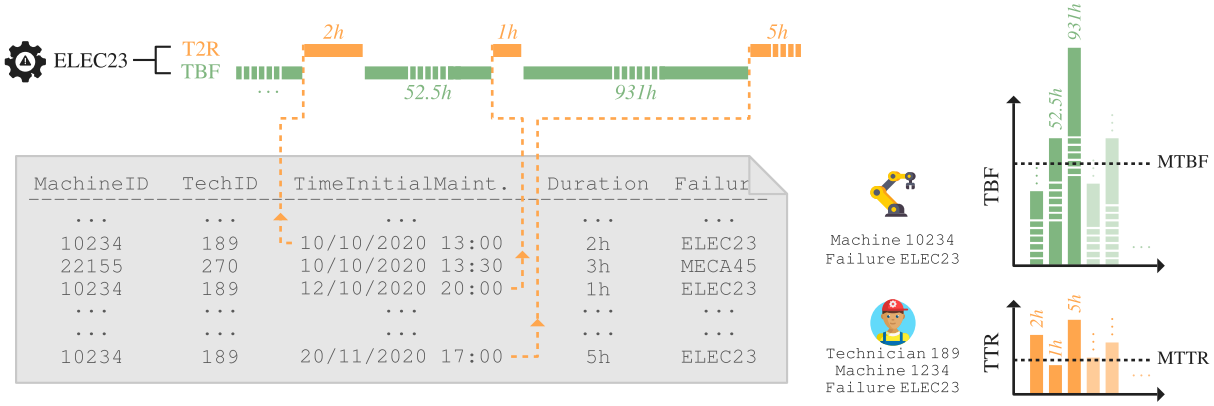


Fig. 6. Illustration of feature extraction methodology based on maintenance data.

Table 6
Comparative performance evaluation of RL, DR, GA, under different levels of uncertainty in Failure Distribution and Maintenance Duration.

Uncertainty		Methods											
Fail Dist	Maint Dur	RL		D1		D2		G1		G2		G4	
		Reward	MTTR	Reward	MTTR	Reward	MTTR	Reward	MTTR	Reward	MTTR	Reward	MTTR
High	Low	27.40	4.03	23.68	6.05	23.99	5.90	27.51	5.83	26.71	5.86	26.56	5.81
Low	Low	44.77	4.54	37.92	6.07	38.65	5.78	46.89	5.80	45.23	5.79	44.70	5.77
High	High	27.19	4.90	25.37	5.96	25.77	5.65	27.64	6.84	27.15	6.60	26.81	6.40
Low	High	45.01	4.74	39.47	6.04	40.28	5.76	46.43	6.71	44.74	6.50	43.95	6.44
Fail Dist	Maint Dur	G6		G8		G12		G16		G20		G24	
		Reward	MTTR	Reward	MTTR	Reward	MTTR	Reward	MTTR	Reward	MTTR	Reward	MTTR
High	Low	26.03	5.64	25.80	5.74	25.25	5.74	23.30	5.88	19.95	5.77	19.06	5.82
Low	Low	44.18	5.62	43.01	5.76	41.44	5.93	37.54	5.93	32.94	5.75	30.73	5.77
High	High	26.42	6.22	26.22	6.48	25.80	6.34	23.96	6.33	20.60	6.15	19.86	5.99
Low	High	43.62	6.33	43.02	6.42	41.54	6.33	38.02	6.33	33.18	6.14	32.26	6.00

Each maintenance entry (ticket) $k \in \mathcal{D}$ is characterized by the following attributes:

- **Creation date and time** (s_k) represents the specific date and time when the maintenance ticket was created.
- **Maintenance Duration** (c_k) denotes the total time taken to complete the maintenance intervention
- **Technician Identifier** (q_k) represents the id of the technician who complete the maintenance intervention
- **Failure type** (f_k) represents the type of failure that was addressed by the technician on the maintenance intervention

All tickets in the maintenance records are chronologically ordered based on their creation date and time s_k.

Using this information, we performed a data processing to obtain the values of μ_{ij}, σ_{ij}, α_{ij}, β_{ij} that will be used to represent the dynamic scheduling environment. The maintenance data is used to calculate the Mean Time to Repair (MTTR) and Mean Time Between Failures (MTBF), as illustrated in Fig. 6. These Key Performance Indicators (KPI) will be used to estimate the failure distribution and to determine the maintenance duration of the technicians per failure.

To model the failure distribution, we subdivided $\mathcal{D}$ by an initial filter into tickets by the type of failure $f \in \mathcal{F}$ and for each technician $j \in \mathcal{T}$, resulting in subsets $\mathcal{D}fj$. Consequently, $\mathcal{D}$ is represented as the union of these subsets, denoted by $\mathcal{D} = \bigcup \mathcal{D}fj$. For each subset $\mathcal{D}fj$, the Time Between Failures (TBF) is determined by $s_{k+1} - (s_k + c_k)$ where $k \in \mathcal{D}_{fj}$ (see green samples in Fig. 6). To model the failure distribution, we fit a two-parameter Weibull distribution using the `Reliability` Library where we subsequently extracted α and β parameters. For the Time to Repair (TTR) for each technician, represented by a normal distribution, we added a second filter in addition to each failure type by filtering for each technician. The TTR for each ticket k was computed based on the duration c_k, with these samples highlighted as orange in Fig. 6. From this procedure, we calculated the average repair time μ_{fj} and the standard deviation σ_{fj} for each failure and technician combination.

4.2. Comparative analysis

In this section, results of the comparative analysis of the methods (RL, GA-S, DR) are presented and discussed. Let us remind ourselves that (i) the performance evaluation of each method is done based on Eq. (4) and considering the MDP defined in Section 3.1. The reader can find hereinafter a few more experimental setting information for each method:

- *RL:* the RL agent was trained using the PPO algorithm (Schulman et al., 2017) based on the SB3 library. For the parameters of PPO, we define the size of the networks for the actor and critic as [256,256] for each. The total number of steps in the environment to train the network is 3M with 250 epochs. The learning rate is defined as a linear schedule from $1e^{-2} \rightarrow 0$ for the total training. Other parameters follow the default settings in SB3. The PPO policy converged after training for 13 h on 4 GPUs of Tesla V100-SXM2-32 GB and 28 CPU cores;
- *GA-S:* the implementation of the GA-S model is based on the `Pymoo` library. We defined different numbers of steps in which the search for the optimal solution will be performed based on current information for the tickets (failures) and the availability of the technicians. The scheduling is performed in every nG step given that $n \in \mathbb{Z}$ and $0 \leq nG < H$;
- *DR:* for the two DRs (FIFO, FIBT), there are no hyperparameters to define. In each timestep, the rules perform an assignment based on current information as described in Section 3.1.

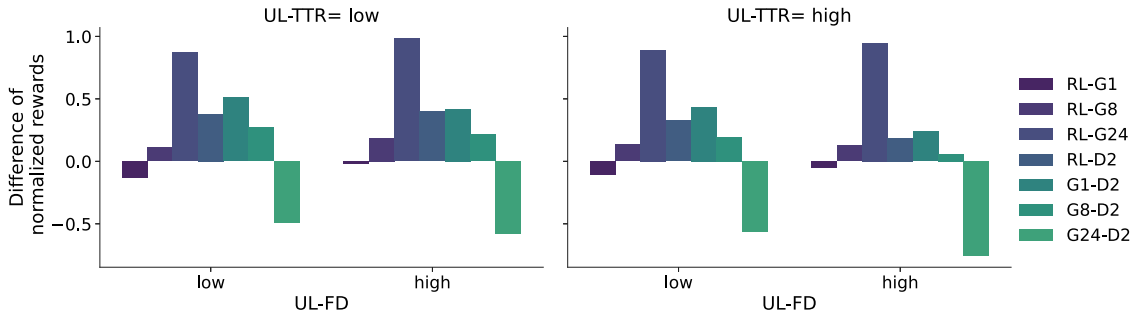


Fig. 7. Performance differential of RL, DR, GA across low and high uncertainty in Time-to-Repair and Failure Distribution.

Table 7
Wall-clock time (s) comparison of RL, GA, and DR for maintenance scheduling under differing degrees of uncertainty.

Configuration		**Methods**					
Failure Distribution	Maintenance Duration	RL	D1	D2	G1	G2	G4
High	Low	0.38	0.05	0.04	6317.05	563.27	226.32
Low	Low	0.35	0.04	0.04	6681.33	606.27	237.62
High	High	0.35	0.04	0.05	6186.50	537.95	226.43
Low	High	0.38	0.04	0.04	6400.39	559.74	229.44
Failure Distribution	Maintenance Duration	G6	G8	G12	G16	G20	G24
High	Low	582.09	417.07	1268.48	917.97	696.99	64.17
Low	Low	598.62	437.04	1270.40	930.85	717.78	72.07
High	High	582.41	416.96	1148.53	872.95	680.05	62.58
Low	High	592.93	418.19	1171.24	894.28	665.13	68.41

The results for the comparison analysis are presented in Table 6 for the three methods.[2] We used the reward and the MTTR as performance indicators as both provide different insights into the performance of the methods in scheduling. The MTTR provides information on the effectiveness of technician allocations in performing interventions, while the reward provides an overall health and productivity of the system as it measure the working state of the machines and the remaining maintenance time. In the case of the reward (Eq. (4)), the results obtained from RL and G1 are quite similar; the total reward obtained is slightly higher in GA-S than in RL for the cases where the uncertainty is low for the failure distribution, but the performance of RL – *compared to GA-S* – increases along with the increase of uncertainty in the failure distribution. This finding can be visualized in Fig. 7 where we display the difference between RL, D2, and GA-S for 1 step (every hour), 8 steps (every shift), and 24 steps (every day). The biggest impact on the reward is influenced by the uncertainty of the Failure Distribution. With the RL approach, the results show that there is a tendency to obtain better results when there is less unpredictability in the Failure Distribution even if there is high uncertainty on how long the maintenance will take. In contrast, the GA approach shows the opposite response under similar conditions. On the other hand, when the uncertainty in the Failure Distribution is high, it is the GA methods that improve the rewards, while RL method's rewards decrease. Regarding DR, even if the methods provide lower rewards, the reward increases when the uncertainty in both Failure Distribution and Maintenance Duration increases. This difference highlights the behavior of the methods under uncertainty and which method might be more reliable under different conditions. Regarding the MTTR, RL showed superior performance, consistently achieving lower values compared to the other methods, including D2, which is responsible for choosing the best technician. This suggests that RL is an effective approach for the maintenance process, leading to a better allocation of resources and a reduction in operational downtime. In the case of DR methods, there is an exceptional behavior for the high uncertainty in the Failure Distribution and the Maintenance Duration, where a decrease is observed in the MTTR for both D1 and D2.

Table 8
Comparative Evaluation of reward, time-to-repair, and solving time for RL, DR, GA methods using real data.

Metric	RL	D1	D2	G1	G2	G4
Reward	11.38	6.83	5.52	6.85	7.06	6.97
MTTR	4.68	4.49	2.62	4.49	4.50	4.56
Time (s)	0.77	0.20	0.19	10 476.30	1061.07	569.52
Metric	G6	G8	G12	G16	G20	G24
Reward	6.76	6.78	6.66	6.57	6.15	6.62
MTTR	4.53	4.51	4.47	4.49	4.23	4.37
Time (s)	1203.02	887.39	2022.17	1580.30	1873.02	224.70

Let us now analyze the wallclock time it takes to make inferences for each of the methods, whose results are given in Table 7 (note: inferences were run on a MacOS Ventura with 2.6 GHz 6-Core Intel Core i7 on the processor and 32 GB 2667 MHz DDR4 on RAM). It can be observed that the method that obtains the results in the shortest time is DR, followed by RL, and finally GA-S. The wallclock time taken by GA-S to obtain the results for each step (1 h step) is higher (≈ 1.72 on average) than the time needed to calculate the steps (1 h), which makes it not applicable in real-time settings with our current computational resources. We should mention that there is no linear relationship between reducing the number of steps and the complexity of the problem when applying GA-S to this problem. When we perform the scheduling in each time step, we have to perform 168 schedules that represent 168 h (equivalent to 1 week); however, the number of interventions to be performed will be significantly reduced because in each schedule we solve certain active tickets. On the other side, reducing the frequency of scheduling involves greater complexity by having more active interventions due to the cumulative tickets that are being generated in each step of time for which we do not perform any action. This effect can be seen as the computational time decreases in G1, G2, and G4, but increases in G6, and increases again from G8 to G12.

[2] We remind the reader that G1, G2, G4,..., G24 refer to the use of GA-S considering different re-optimization periods for generating the re-scheduling (i.e., every hour, 2 h, 4 h,..., 24 h).

4.3. Use case study of a manufacturing industry

We present here the real-life case study using 10 years of historical maintenance data from a manufacturing company. To do so, we conducted the analysis described in Section 4.1.3, which resulted in the derivation of 100 Weibull distributions for all types of failures and machines and 8 technicians. For this, ten representative Weibull distributions were chosen with sufficient samples at the fitting time, and making sure they are satisfying specific criteria. Among other criteria, a shape parameter greater than 1.0 and the distributions with a low scale parameter is expected, which eases the execution of the experiments since the failure distributions happen in a short amount of time modeling a wear-out degradation. Let these ten distributions be represented by lists α_{real} and β_{real}, with ten values each representing the scale and shape parameters of a Weibull distribution, respectively. Our experiments are based on a set of 100 Weibull distributions formed by combining the values of α_{real} and β_{real} defined as:

$$\{W(s,k) : s \in \alpha_{real}, k \in \beta_{real}\}$$

The results, presented in Table 8, reveal important insights of the behavior of the scheduling methods for the real-based scenario.

- **GA-S** shows a minimal variation for the different reschedule frequency, suggesting that the occurrence of failures is prolonged. Consequently, having a lower frequency reschedule (e.g. 24 h reschedule) offers optimal benefits in terms of wallclock time, as frequent rescheduling does not provide significantly benefits.
- **RL** once trained, shows a shorter inference time compared to GA-S. This efficiency translates into significant time savings when used recurrently, as illustrated in Fig. 8. Furthermore, the results presented show that the reward is considerably higher compared to the other methods indicating how well RL methods deal with real data distributions to keep machines in a working state.
- **DR** present lower reward than comparing with the best model of the other methods. However, the D2 variant of DR effectively reduces MTTR by assigning the best technicians compared to GA and RL. This trade-off between suggests that although D2 optimizes technician assignment, it may not adequately address the complexity of the environment, particularly in prioritizing less frequent but critical failures.

Based on these findings, selecting the appropriate scheduling method involves a strategic decision. While GA-S may be less efficient in terms of computational time, its consistent results across rescheduling frequencies is suitable for a more stable and less complex parameter tuning, in addition, it presents a well-balanced solution between the reward and MTTR. In contrast, the RL approach is highly time efficient and generates higher rewards, making it attractive for operations that need high uptime. Lastly, rule-based methods are an alternative without the need for parameter selection that demonstrates excellent performance in minimizing the MTTR with a low computational time. Each of these methods presents benefits and challenges that the decision maker should consider.

4.4. Financial consideration

The objective of the proposed model is to minimize the downtime of the equipment which leads to the reduction of the maintenance duration. It should be noted that several costs are involved in the maintenance operations, including e.g. the type of maintenance performed. Imperfect maintenance allows to restore the equipment to a previous state by extending the RUL of the component or equipment, or replacement operation, whose purpose is to restore the equipment as good as new by substituting the component by a new (or almost new) one. The cost of these different types of maintenance can vary significantly, with maintenance replacement operations generally being more expensive. In addition, if the component of an equipment cannot receive maintenance because it reaches its useful life, a spare part needs to be installed. This is one of the most significant costs after a breakdown especially for high-end equipment. Another factor to consider is the cost of the workforce. Indeed, maintenance operations require skilled technicians who can perform maintenance with different degrees of quality, not only by reducing the MTTR but also maximizing the MTBF. On the business side, a decision maker needs to consider the advantage of extending the lifespan of the equipment to keep a high production against the cost of performing maintenance considering spare parts, technicians, type of maintenance, among other factors.

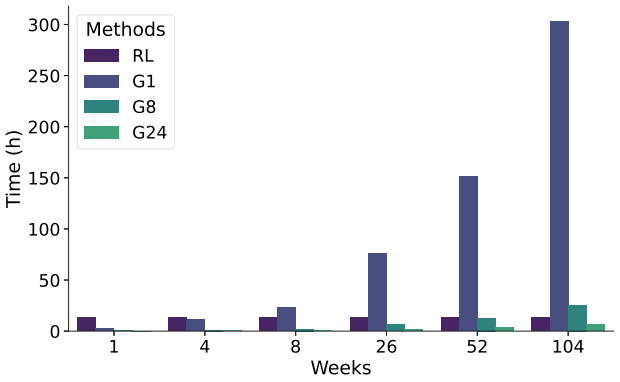


Fig. 8. Wall-Clock time required by RL, G1, G8, G24 for scheduling over different periods of time.

5. Conclusion and future work

Maintenance scheduling implies to deal with different types of uncertainties, spanning from the maintenance duration (which may highly vary depending on the assigned technician) to RUL estimation/approximation or unexpected changes in the production schedule. While metaheuristics and some extensions (e.g., Simheuristics) have been widely used to deal with stochastic combinatorial optimization problems, Reinforcement Learning is increasingly used in all sectors due to democratization of AI/ML.

To the best of our knowledge, no research study has ever analyzed how such methods – *RL, GA-simheuristic and traditional dispatching rules (DR) in the present paper* – perform when faced with different levels of uncertainty in the scheduling problem. The present paper overcomes this lack of study by considering two types of uncertainty as part of the stochastic scheduling problem (uncertainty about machines' failure distribution and time to repair of technicians), with the overall goal to increase machine uptime and reduce mean time to repair, while minimizing wallclock time. Experiments are carried out considering both artificial/synthetic data and a real-life use case with a manufacturing company (using 10 years of maintenance data). The experimental results indicate that RL shows an exceptional adaptability to decrease the MTTR, especially in the face of high uncertainty. Overall, GA-S performs well in several scenarios, particularly when the re-optimization frequency is high (re-optimized every hour), but has the disadvantage of having a high inference time. Finally, DR policies provide a computationally efficient solution for fast decision making under resource constrained conditions. For example, in the case of the real-life use case, DRs obtain high performance in reducing the time to repair compared to RL and GA-S. Although the exploration space is large, the complexity of the environment is low, managing to outperform in terms of MTTR.

In future work, we plan to consider hybrid approaches to take advantage of the strengths of the different methods (e.g., computational efficiency that can reflect the carbon footprint) while mitigating their weaknesses. In addition, we will investigate uncertainty as a continuous parameter to comprehensively assess its different effects and trade-offs.

Finally, we seek to consider the uncertainty of the production processes and machine degradation to perform an opportunistic scheduling and look for the best time window to optimize the MTTR, but also to maintain a high production that turns into economic benefit.

CRediT authorship contribution statement

Marcelo Luis Ruiz-Rodríguez: Conceptualization, Methodology, Software, Writing – original draft, Writing – review & editing. **Sylvain Kubler:** Conceptualization, Methodology, Writing – original draft, Writing – review & editing, Supervision. **Jérémy Robert:** Conceptualization, Methodology, Supervision. **Yves Le Traon:** Conceptualization, Supervision.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Marcelo Luis Ruiz Rodriguez reports financial support was provided by Luxembourg National Research Fund. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This research was funded in whole or in part by the Luxembourg National Research Fund (FNR), grant reference 16756339. For the purpose of open access, and in fulfillment of the obligations arising from the grant agreement, the author has applied a Creative Commons Attribution 4.0 International (CC BY 4.0) license to any Author Accepted Manuscript version arising from this submission.

References

Alves, F. F., & Ravetti, M. G. (2020). Hybrid proactive approach for solving maintenance and planning problems in the scenario of Industry 4.0. *IFAC-PapersOnLine*, *53*(3), 216–221. http://dx.doi.org/10.1016/j.ifacol.2020.11.035.

Arena, M., Di Pasquale, V., Iannone, R., Miranda, S., & Riemma, S. (2022). A maintenance driven scheduling cockpit for integrated production and maintenance operation schedule. *Advances in Manufacturing*, *10*(2), 205–219. http://dx.doi.org/10.1007/s40436-021-00380-z.

Benaggoune, K., Meraghni, S., Ma, J., Mouss, L. H., & Zerhouni, N. (2020). Post prognostic decision for predictive maintenance planning with remaining useful life uncertainty. In *Proceedings - 2020 prognostics and health management conference* (pp. 194–199). http://dx.doi.org/10.1109/PHM-Besancon49106.2020.00039.

Bencheikh, G., Letouzey, A., & Desforges, X. (2022). An approach for joint scheduling of production and predictive maintenance activities. *Journal of Manufacturing Systems*, *64*, 546–560. http://dx.doi.org/10.1016/j.jmsy.2022.08.005.

Celen, M., & Djurdjanovic, D. (2020). Integrated maintenance and operations decision making with imperfect degradation state observations. *Journal of Manufacturing Systems*, *55*, 302–316. http://dx.doi.org/10.1016/j.jmsy.2020.03.010.

De Jonge, B., Klingenberg, W., Teunter, R., & Tinga, T. (2015). Optimum maintenance strategy under uncertainty in the lifetime distribution. *Reliability Engineering & System Safety*, *133*, 59–67. http://dx.doi.org/10.1016/j.ress.2014.09.013.

De Jonge, B., & Scarf, P. A. (2020). A review on maintenance optimization. *European Journal of Operational Research*, *285*(3), 805–824. http://dx.doi.org/10.1016/j.ejor.2019.09.047.

Deng, S., Zhu, Y., Yu, Y., & Huang, X. (2024). An integrated approach of ensemble learning methods for stock index prediction using investor sentiments. *Expert Systems with Applications*, *238*, Article 121710. http://dx.doi.org/10.1016/j.eswa.2023.121710.

Detti, P., Nicosia, G., Pacifici, A., & Zabalo Manrique de Lara, G. (2019). Robust single machine scheduling with a flexible maintenance activity. *Computers & Operations Research*, *107*, 19–31. http://dx.doi.org/10.1016/j.cor.2019.03.001.

Ferjani, A., Ammar, A., Pierreval, H., & Elkosantini, S. (2017). A simulation–optimization based heuristic for the online assignment of multi-skilled workers subjected to fatigue in manufacturing systems. *Computers & Industrial Engineering*, *112*, 663–674. http://dx.doi.org/10.1016/j.cie.2017.02.008.

Geurtsen, M., Didden, J. B. H. C., Adan, J., Atan, Z., & Adan, I. (2023). Production, maintenance and resource scheduling: A review. *European Journal of Operational Research*, *305*(2), 501–529. http://dx.doi.org/10.1016/j.ejor.2022.03.045.

Ghaleb, M., Taghipour, S., Sharifi, M., & Zolfagharinia, H. (2020). Integrated production and maintenance scheduling for a single degrading machine with deterioration-based failures. *Computers & Industrial Engineering*, *143*, Article 106432. http://dx.doi.org/10.1016/j.cie.2020.106432.

Guan, S., Zhuang, Z., Tao, H., Chen, Y., Stojanovic, V., & Paszke, W. (2023). Feedback-aided PD-type iterative learning control for time-varying systems with non-uniform trial lengths. *Transactions of the Institute of Measurement and Control*, *45*(11), 2015–2026. http://dx.doi.org/10.1177/01423312221142564.

Huang, J., Chang, Q., & Arinez, J. (2020). Deep reinforcement learning based preventive maintenance policy for serial production lines. *Expert Systems with Applications*, *160*, Article 113701. http://dx.doi.org/10.1016/j.eswa.2020.113701.

Jayatilleka, S., & McLinn, J. (2021). Practical implications of Weibull shape parameter; lessons & pitfalls. In *2021 annual reliability and maintainability symposium* (pp. 1–6). http://dx.doi.org/10.1109/RAMS48097.2021.9605732.

Kaufmann, E., Bauersfeld, L., Loquercio, A., Müller, M., Koltun, V., & Scaramuzza, D. (2023). Champion-level drone racing using deep reinforcement learning. *Nature*, *620*(7976), 982–987. http://dx.doi.org/10.1038/s41586-023-06419-4.

Kuhnle, A., Jakubik, J., & Lanza, G. (2019). Reinforcement learning for opportunistic maintenance optimization. *Production Engineering*, *13*(1), 33–41. http://dx.doi.org/10.1007/s11740-018-0855-7.

Lu, B., Chen, Z., & Zhao, X. (2021). Data-driven dynamic predictive maintenance for a manufacturing system with quality deterioration and online sensors. *Reliability Engineering & System Safety*, *212*, Article 107628. http://dx.doi.org/10.1016/j.ress.2021.107628.

Lutz, I. D., Wang, S., Norn, C., Courbet, A., Borst, A. J., Zhao, Y. T., Dosey, A., Cao, L., Xu, J., Leaf, E. M., Treichel, C., Litvicov, P., Li, P., Goodson, A. D., Rivera-Sánchez, P., Bratovianu, A. M., Baek, M., King, N. P., Ruohola-Baker, H., & Baker, D. (2023). Top-down design of protein architectures with reinforcement learning. *Science*, *380*(6642), 266–273. http://dx.doi.org/10.1126/science.adf6591.

Mi, S., Feng, Y., Zheng, H., Li, Z., Gao, Y., & Tan, J. (2020). Integrated intelligent green scheduling of predictive maintenance for complex equipment based on information services. *IEEE Access*, *8*, 45797–45812. http://dx.doi.org/10.1109/ACCESS.2020.2977667.

Miao, B., Deng, Q., Zhang, L., Huo, Z., & Liu, X. (2022). Collaborative scheduling of spare parts production and service workers driven by distributed maintenance demand. *Journal of Manufacturing Systems*, *64*, 261–274. http://dx.doi.org/10.1016/j.jmsy.2022.06.012.

Nasruddin, Nasution, S., Aisyah, N., Surachman, A., & Wibowo, A. S. (2018). Exergy analysis and exergoeconomic optimization of a binary cycle system using a multi objective genetic algorithm. *International Journal of Technology*, *9*(2), 275–286. http://dx.doi.org/10.14716/ijtech.v9i2.1040.

Németh, I., Kocsis, Á., Takács, D., Shaheen, B. W., Takács, M., Merlo, A., Eytan, A., Bidoggia, L., & Olocco, P. (2020). Maintenance schedule optimisation for manufacturing systems. *IFAC-PapersOnLine*, *53*(3), 319–324. http://dx.doi.org/10.1016/j.ifacol.2020.11.051.

Neumann, A., Gounder, S., Yan, X., Sherman, G., Campbell, B., Guo, M., & Neumann, F. (2023). Diversity optimization for the detection and concealment of spatially defined communication networks. In *GECCO 2023 - proceedings of the 2023 genetic and evolutionary computation conference* (pp. 1436–1444). http://dx.doi.org/10.1145/3583131.3590405.

Pan, W., & Liu, S. Q. (2023). Deep reinforcement learning for the dynamic and uncertain vehicle routing problem. *Applied Intelligence*, *53*(1), 405–422. http://dx.doi.org/10.1007/s10489-022-03456-w.

Rokhforoz, P., & Fink, O. (2021). Distributed joint dynamic maintenance and production scheduling in manufacturing systems: Framework based on model predictive control and benders decomposition. *Journal of Manufacturing Systems*, *59*, 596–606. http://dx.doi.org/10.1016/j.jmsy.2021.04.010.

Ruiz Rodríguez, M. L., Kubler, S., de Giorgio, A., Cordy, M., Robert, J., & Le Traon, Y. (2022). Multi-agent deep reinforcement learning based predictive maintenance on parallel machines. *Robotics and Computer-Integrated Manufacturing*, *78*, Article 102406. http://dx.doi.org/10.1016/j.rcim.2022.102406.

Ruschel, E., Santos, E. A. P., & Loures, E. de F. R. (2020). Establishment of maintenance inspection intervals: an application of process mining techniques in manufacturing. *Journal of Intelligent Manufacturing*, *31*(1), 53–72. http://dx.doi.org/10.1007/s10845-018-1434-7.

Sarazin, A., Bascans, J., Sciau, J. B., Song, J., Supiot, B., Montarnal, A., Lorca, X., & Truptil, S. (2021). Expert system dedicated to condition-based maintenance based on a knowledge graph approach: Application to an aeronautic system. *Expert Systems with Applications*, *186*, Article 115767. http://dx.doi.org/10.1016/j.eswa.2021.115767.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal policy optimization algorithms. https://arxiv.org/abs/1707.06347.

Song, X., Sun, P., Song, S., & Stojanovic, V. (2023). Finite-time adaptive neural resilient DSC for fractional-order nonlinear large-scale systems against sensor-actuator faults. *Nonlinear Dynamics*, *111*(13), 12181–12196. http://dx.doi.org/10.1007/s11071-023-08456-0.

Stojanovic, V. (2023). Fault-tolerant control of a hydraulic servo actuator via adaptive dynamic programming. *Mathematical Modelling and Control*, *3*(3), 181–191. http://dx.doi.org/10.3934/mmc.2023016.

Su, J., Huang, J., Adams, S., Chang, Q., & Beling, P. A. (2022). Deep multi-agent reinforcement learning for multi-level preventive maintenance in manufacturing systems. *Expert Systems with Applications*, *192*, Article 116323. http://dx.doi.org/10.1016/j.eswa.2021.116323.

Sulewski, P., & Szymkowiak, M. (2022). The Weibull lifetime model with randomised failure-free time. *Statistics in Transition New Series*, *23*(4), 59–76. http://dx.doi.org/10.2478/stattrans-2022-0042.

Sun, Q., Ye, Z. S., & Peng, W. (2019). Scheduling preventive maintenance considering the saturation effect. *IEEE Transactions on Reliability*, *68*(2), 741–752. http://dx.doi.org/10.1109/TR.2018.2874265.

Valet, A., Altenmüller, T., Waschneck, B., May, M. C., Kuhnle, A., & Lanza, G. (2022). Opportunistic maintenance scheduling with deep reinforcement learning. *Journal of Manufacturing Systems*, *64*, 518–534. http://dx.doi.org/10.1016/j.jmsy.2022.07.016.

Wocker, M., Betz, N. K., Feuersänger, C., Lindworsky, A., & Deuse, J. (2020). Unsupervised learning for opportunistic maintenance optimization in flexible manufacturing systems. *Procedia CIRP*, *93*, 1025–1030. http://dx.doi.org/10.1016/j.procir.2020.04.025.

Xia, T., Shi, G., Si, G., Du, S., & Xi, L. (2021). Energy-oriented joint optimization of machine maintenance and tool replacement in sustainable manufacturing. *Journal of Manufacturing Systems*, *59*, 261–271. http://dx.doi.org/10.1016/j.jmsy.2021.01.015.

Yan, Q., Wang, H., & Wu, F. (2022). Digital twin-enabled dynamic scheduling with preventive maintenance using a double-layer Q-learning algorithm. *Computers & Operations Research*, *144*, Article 105823. http://dx.doi.org/10.1016/j.cor.2022.105823.

Yang, C. Y., Shiranthika, C., Wang, C. Y., Chen, K. W., & Sumathipala, S. (2023). Reinforcement learning strategies in cancer chemotherapy treatments: A review. *Computer Methods and Programs in Biomedicine*, *229*, Article 107280. http://dx.doi.org/10.1016/j.cmpb.2022.107280.

Yazdani, R., Alipour-Vaezi, M., Kabirifar, K., Salahi Kojour, A., & Soleimani, F. (2022). A lion optimization algorithm for an integrating maintenance planning and production scheduling problem with a total absolute deviation of completion times objective. *Soft Computing*, *26*(24), 13953–13968. http://dx.doi.org/10.1007/s00500-022-07436-7.

Ying, K. C., Lu, C. C., & Chen, J. C. (2016). Exact algorithms for single-machine scheduling problems with a variable maintenance. *Computers & Industrial Engineering*, *98*, 427–433. http://dx.doi.org/10.1016/j.cie.2016.05.037.

Yu, T., Zhu, C., Chang, Q., & Wang, J. (2019). Imperfect corrective maintenance scheduling for energy efficient manufacturing systems through online task allocation method. *Journal of Manufacturing Systems*, *53*, 282–290. http://dx.doi.org/10.1016/j.jmsy.2019.11.002.

Zahir, A. A. M., Alhady, S. S. N., Othman, W. A. F. W., Wahab, A. A. A., & Ahmad, M. F. (2020). Objective functions modification of GA optimized PID controller for brushed DC motor. *International Journal of Electrical and Computer Engineering*, *10*(3), 2426–2433. http://dx.doi.org/10.11591/ijece.v10i3.pp2426-2433.

Zhuang, Z., Tao, H., Chen, Y., Stojanovic, V., & Paszke, W. (2023). An optimal iterative learning control approach for linear systems with nonuniform trial lengths under input constraints. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *53*(6), 3461–3473. http://dx.doi.org/10.1109/TSMC.2022.3225381.