PhD-FSTM-2024-098
The Faculty of Science, Technology and Medicine

DISSERTATION

Defence held on 21/11/2024 in Esch-sur-Alzette

to obtain the degree of

DOCTEUR DE L'UNIVERSITÉ DU LUXEMBOURG
EN INFORMATIQUE

by

Marcelo Luis RUIZ RODRÍGUEZ
Born on 13^{th} August 1990 in Nuevo León (Mexico)

Maintenance optimization in Industry 4.0: A Deep Reinforcement Learning Approach to Sustainable Policy Development

Dissertation defence committee

Dr. Yves Le Traon, Dissertation Supervisor
Professor, Université du Luxembourg

Dr. Nicolas Navet, Chairman
Professor, Université du Luxembourg

Dr. Hind Bril El Haouzi, Vice Chairman
Professor, Université de Lorraine

Dr. Damien Trentesaux
Professor, Université Polytechnique Hauts-de-France

Dr. Sylvain Kubler
Senior Researcher, Université du Luxembourg

UNIVERSITÉ DU LUXEMBOURG

PhD-FSTM-2024-098
The Faculty of Science, Technology and Medicine

DISSERTATION

Defence held on 21/11/2024 in Esch-sur-Alzette

to obtain the degree of

DOCTEUR DE L'UNIVERSITÉ DU LUXEMBOURG
EN INFORMATIQUE

by

Marcelo Luis RUIZ RODRÍGUEZ
Born on 13^{th} August 1990 in Nuevo León (Mexico)

Maintenance optimization in Industry 4.0: A Deep Reinforcement Learning Approach to Sustainable Policy Development

Dissertation defence committee

Dr. Yves Le Traon, Dissertation Supervisor
Professor, Université du Luxembourg

Dr. Nicolas Navet, Chairman
Professor, Université du Luxembourg

Dr. Hind Bril El Haouzi, Vice Chairman
Professor, Université de Lorraine

Dr. Damien Trentesaux
Professor, Université Polytechnique Hauts-de-France

Dr. Sylvain Kubler
Senior Researcher, Université du Luxembourg

Dr. Jérémy Robert
Digital Technology Specialist, Cebi International S.A.

Abstract

Effective maintenance planning and scheduling are essential for manufacturing companies to prevent machine breakdowns and maximize uptime and production. Furthermore, these policies must be in alignment with the principles of environmental integrity and social responsibility. The development of sustainable policies presents several challenges. These include the need to balance economic, environmental, and social aspects, as well as to address uncertainties such as unexpected failures and variable time-to-repair (TTR). This thesis, conducted in partnership with Cebi, an electromechanical component design and manufacturing company, addresses the challenge of developing sustainable maintenance policies in the face of uncertainty. To this end, we propose a Deep Reinforcement Learning (DRL)-based approach for predictive maintenance, which we compare with traditional maintenance policies such as corrective, preventive, and condition-based maintenance, and evaluate against diverse methods based on metaheuristics and rule-based approaches.

As a first contribution, we conducted a study of the impact of categorized levels of uncertainty in a manufacturing environment on the failure distribution and time to repair for maintenance policies. We evaluated the performance of DRL, genetic algorithm-based simheuristic (GA-S), and rule-based (DR) decision-making systems in terms of mean time to repair (MTTR), machine uptime, and computational efficiency. The work was conducted in simulated scenarios with different levels of uncertainty and also considering a real manufacturing use case. The experimental results show that DRL shows exceptional adaptability to reduce MTTR, especially in the face of high uncertainty. GA-S outperforms DRL and DR in terms of total machine uptime, but not in terms of MTTR, when configured with high re-optimization frequencies (i.e., hourly re-optimization), but rapidly underperforms as the re-optimization frequency decreases. Furthermore, our study shows that GA-S is computationally expensive compared to DRL and DR policies.

As a second contribution, we proposed to tackle the problem of maintenance scheduling on multi-component identical parallel machines performed by technicians. In this work, we proposed a multi-agent DRL approach to learn the maintenance policy under the uncertainty of multiple machine failures. This approach comprises

DRL agents that partially observe the state of each machine to coordinate the decision-making in maintenance scheduling, resulting in the dynamic assignment of maintenance tasks to technicians (with different skills) over a set of machines. Experimental evaluation shows that our DRL-based maintenance policy outperforms classical maintenance policies such as Corrective Maintenance (CM) and Preventive Maintenance (PM) in terms of failure prevention and downtime, improving overall performance by $\approx 75\%$.

Our last contribution proposed to optimize maintenance scheduling from an economic perspective (considering maintenance, breakdown, and downtime costs), an environmental perspective (considering the carbon footprint produced during production) and a social perspective (considering the fatigue experienced by technicians during maintenance activities). We propose an evolutionary multi-objective multi-agent deep Q-Learning (EvoDQN)-based approach, where multiple agents explore the preference space to maximize the hypervolume of these sustainable objectives. The results demonstrate the trade-offs between these objectives when compared to traditional maintenance policies such as condition-based maintenance (CBM) and CM, as well as different deep Q-network (DQN) policies trained with various preferences and a higher number of agents for our EvoDQN approach. Finally, our approach evaluated the production cycle, in which our method demonstrated superior performance, resulting in increased profitability within the system.

Acknowledgments

I would like to express my sincerest gratitude to all those who have supported and guided me throughout my doctoral studies. This thesis would not have been possible without their invaluable contributions, both academic and personal.

I would like to express my deepest gratitude to Prof. Dr. Yves Le Traon, who gave me the opportunity to be my supervisor and to join the SERVAL group. I feel incredibly fortunate and honored to have worked under his guidance. His expertise and positive spirit provided me with the intellectual and personal support to continue this adventure. My gratitude extends to Dr. Mike Papadakis for providing insights and challenging questions that contributed to this doctoral work. I'm really thankful to Dr. Maxime Cordy, with whom I had the opportunity to participate in various scientific discussions and reading groups that enriched my knowledge during this work. I am especially grateful to Dr. Sylvain Kubler, with whom I worked on a daily basis and who always had confidence in me and gave me the freedom to choose the path I wanted to take in my doctoral work. Beyond guiding me through different aspects of academia, he gave me all the personal support for which I am deeply grateful. I would like to extend my heartfelt thanks to Solène Vincens and Fiona Levasseur, who provided me with all the necessary administrative assistance during my doctoral journey as well as to all the staff of the University of Luxembourg and SnT.

I would like to express my gratitude to CEBI, especially to Guillaume Policarpo, Dr. Jérémy Robert, and Dr. Paul-Lou Benedick, who not only gave me the opportunity to work with them but also granted me the freedom to follow my own course during this doctoral work.

I would also like to express my deeply gratitude to all my friends and colleagues at SERVAL, with whom I have shared numerous joys and challenges and with whom we shared the same destiny following different paths.

I would like to thank the Luxembourg National Research Fund (FNR) for providing me with the financial support to carry out this doctoral work.

Finally, I wish to express my deep gratitude to my family, (to be Dr.)Aurora, Carolina and Julio, for their unconditional support throughout this adventure. I love you with all my heart.

Contents

1

Introduction

In this chapter we introduce what is the problem of optimizing maintenance scheduling and the challenges involved. Finally, we highlight what contributions are present in this thesis.

Contents

1.1 Context

Industry 4.0 (I4.0) technologies are transforming the business models of manufacturing companies, enabling greater production flexibility, efficiency, and productivity [Moh18]. Thanks to these technologies, the industry has matured in its maintenance processes from CM, where maintenance is performed in a reactive manner, to Predictive Maintenance (PdM), where real-time monitoring of machines or production lines makes it possible to detect when a machine is likely to fail and to intervene before these failures occur by assigning maintenance personnel with the qualifications to perform these maintenance actions.

Maintenance scheduling is a critical aspect of any business, with approaches that vary based on industry-specific needs and capabilities. Static and dynamic scheduling represent two distinct approaches to this process. In static scheduling, all maintenance activities are planned in advance, taking into account their estimated completion times, the resources needed to complete them (including personnel, tools, and parts to be repaired or replaced), and the time and location of the maintenance activities. In contrast, dynamic scheduling assigns maintenance activities in real time as needed, offering a more efficient and optimal response to resource allocation, greater resilience to sudden changes, and the ability to adjust to new interventions or unexpected delays.

Maintenance is an essential component of any production system as it ensures reliable and efficient operation. Not only does it maintain operational functionality, but also optimizes the timing of interventions by assigning the most appropriate personnel, thereby optimizing the TTR. Effective maintenance strategies help to prevent multiple machine breakdowns and avoid resource conflicts, such as when a technician is assigned to a machine while other machines are down waiting for service.

Maintenance scheduling requires careful coordination of several factors, including technician availability, production schedules, and the potential for unexpected equipment failures. The unpredictability of machine failures and the need to minimize downtime introduce additional complexity to the scheduling process.

There are a variety of optimization techniques that can be employed for the purpose of maintenance optimization. These can be grouped into three main categories. The first of these is Mathematical Programming (MP), which formalizes the maintenance problem as a mathematical model. This allows the type of problem to be defined, whether it is a cost or time minimization problem, or a problem that aims to maximize the reliability or operation of the equipment. These problems require the definition of various constraints, including the non-overlapping of assigned technicians, the order of maintenance operations, and the restriction of the desired budget. The second is Metaheuristic-based methods, that are designed to quickly identify a suitable solution, even for highly complex

or large-scale problems. These methods are typically applied to problems with a large search space or nonlinear constraints. In metaheuristic approaches, the maintenance scheduling problem is typically transformed or reformulated in a way that aligns with the chosen metaheuristic algorithm, ensuring it can explore potential solutions effectively. For instance, Ant Colony Optimization (ACO), methods adapt a scheduling problem to nodes (maintenance activities) where the ants optimize the best route (where the edges can be resources) to form these schedules. Other methods, such as Genetic Algorithms (GA), encode the problem in a chromosome where each gene corresponds to a certain activity and its position in the chromosome indicates the order of the activity to be performed. Finally, there are Machine Learning (ML) based methods using supervised, unsupervised or reinforcement learning approaches (RL). In supervised learning, the use of historical data can allow to recognize patterns that allow to detect when an equipment is more likely to fail and to be able to assign maintenance activities. In unsupervised learning, anomalies can be identified in the equipment, especially those that are of the same type, in order to be able to perform interventions. Lastly, in RL, an agent must learn a policy through trial and error in an uncertain environment that is influenced by the agent's actions. The environment provides a reward signal that indicates which actions benefit the agent's behavior and which actions are better avoided. This type of method allows for the creation of dynamic and adaptive schedules by interacting with the environment with different levels of uncertainty. All acronyms used in this thesis are summarized in Table 1.1.

1.2 Challenges

There are several challenges to optimizing maintenance in the manufacturing industry that have not been properly addressed.

1. **Uncertainty in production and maintenance processes** [DS20]. During production on the shop floor, machines go through degradation processes that cause interruptions that affect scheduling and production times. These stoppages are caused by machines or components reaching the end of their useful life. Depending on the production process, machine type or operating conditions, different failure distributions occur [SBS+21]. In addition, the repair process to restore the machine to a previous or like-new condition can present a variability in repair time due to the complexity of the failure type, technician experience, and skill [FAP+17] .

2. **Designing adaptive and effective maintenance policies** [VAW+22]. Traditional maintenance policies, such as corrective and preventive maintenance, have been the standard for many years. However, these strate-

gies are too rigid to effectively address the dynamic and unpredictable nature of manufacturing environments [ZZ17; YTC22]. These policies must be able to dynamically adapt to changing machine conditions [AX17], varying technician skill levels and availability [TDF+21], and changing organizational priorities [GS18], all while effectively managing uncertainty. The complexity of these requirements makes it difficult to create a one-size-fits-all solution, so it is necessary to develop policies that can learn and adapt in real time to optimize machine uptime and reduce unnecessary interventions.

3. **Sustainability in the maintenance scheduling** [SDL20].
Sustainability has emerged as a major priority for businesses across various industries, and the manufacturing sector is no exception. Production and maintenance processes now need to be economically profitable while also adopting practices that adhere to the principles of environmental integrity and social responsibility [JLK20]. Defining a maintenance policy that takes into account social and/or environmental aspects can lead to a decrease in profitability, but it allows to comply with the standards and demands of the government and customers regarding the importance of the environment, such as energy and carbon footprint reduction [XSS+21; AAA+24], and social responsibility to maintain safe and healthy conditions and improve employee productivity [SCO+20].

1.3 Overview of the contribution

This disseration address the above challenges and brings three contributions.

Dynamic maintenance scheduling approach under uncertainty: Comparison between reinforcement learning, genetic algorithm simheuristic, dispatching rules (Chapter 4) To address the first challenge, we analyze the impact of categorized levels of uncertainty, specifically high and low, on the failure distribution and time to repair. The experiments are conducted in simulated scenarios with different levels of uncertainty and also considering a real manufacturing use case. The results show that rescheduling based on a GA-simheuristic outperforms RL and DR in terms of total machine uptime, but not in terms of mean time to repair, when configured with high re-optimization frequencies (i.e., hourly re-optimization), but rapidly underperforms as the re-optimization frequency decreases. We also show that the GA-S is computationally expensive compared to RL and DR policies.

Multi-agent Deep Reinforcement Learning based Predictive Maintenance on Parallel Machines (Chapter 5). To address the second challenge, we proposes a multi-agent approach that learns a maintenance policy performed

by technicians, under the uncertainty of multiple machine failures. This approach comprises RL agents that partially observe the state of each machine to coordinate the decision-making in maintenance scheduling, resulting in the dynamic assignment of maintenance tasks to technicians (with different skills) over a set of machines. Experimental evaluation shows that our RL-based maintenance policy outperforms traditional maintenance policies (incl., CM and PM) in terms of failure prevention and downtime, improving by $\approx 75\%$ the overall performance.

Evolutionary multi-objective multi-agent reinforcement learning for sustainable maintenance scheduling (Chapter 6). To address the third challenge, we explored an innovative approach aimed at optimizing maintenance scheduling from an economic perspective (considering maintenance, breakdown, downtime costs), an environmental perspective (considering the carbon footprint produced during production), and a social perspective (considering the fatigue experienced by technicians during maintenance activities). To the best of our knowledge, this is the first study to propose a manufacturing scheduling approach that considers all three pillars of sustainability. Another significant contribution of this research is the innovative way in which the optimization problem is addressed. We propose an evolutionary multi-objective multi-agent deep Q-Learning based approach (EvoDQN), where multiple agents explore the preference space to maximize the hypervolume of these sustainable objectives. The results demonstrate the trade-offs between these objectives when compared to traditional maintenance policies such as CM and CBM, as well as different DQN policies trained with various preferences. Our approach demonstrates superior performance, resulting in increased profitability within the system.

Table 1.1: List of acronyms

Acronym	**Definition**
AD	Artificial Data
AI	Artificial Intelligence
CBM	Condition-Based Maintenance
CM	Corrective Maintenance
CMMS	Computerized Maintenance Management System
CNC	Computer Numerical Computation
De	Deterministic
DQN	Deep Q-Network
DR	Dispatching Rules
Dy	Dynamic
Eco	Economic
Env	Environmental
EvoDQN	Evolutionary Multiobjective Multiagent DQN
FIFO	First-In-First-Out
FIBT	First-In-Best-Technician
GA	Genetic Algorithm
KPI	Key Performance Indicators
MDP	Markov Decision Process
Me	Metaheuristic
MG	Markov Game
MILP	Mixed Integer Linear Programming
MINLP	Mixed Integer Non-Linear Programming
MIP	Mixed Integer Programming
ML	Machine Learning
MOMDP	Multi-objective Markov Decision Process
MP	Mathematical Programming
MTTR	Mean Time to Repair
NSGA-II	Non-dominated Sorting Genetic Algorithm II
PdM	Predictive Maintenance
PM	Preventive Maintenance
PPO	Proximal Policy Optimization
RD	Real Data
RL	Reinforcement Learning
RUL	Remaining Useful Life
SA	Simulated Annealing
Soc	Social
TBL	Triple Bottom Line
TS	Tabu Search
VNS	Variable Neighborhood Search
XAI	eXplainable Artificial Intelligence

2

Background

This chapter introduces the basic concepts of maintenance in industry and Reinforcement Learning used in this dissertation.

Contents

2.1 Maintenance in Manufacturing

The industry has benefited from technologies such as cyber-physical systems, internet of things, big data analytics, or still Artificial Intelligence (AI) [SML+21], with the overall aim of increasing profitability, production, capacity, quality, employee safety, and decreasing costs [CA21]. Such technologies are widely used when it comes to maintenance operations, which is the practice of keeping equipment in good operational condition and ensuring that its functions are performed efficiently. In some cases, maintenance can represent up to 60% of the total turnover [Kom02; ZdCd+20].

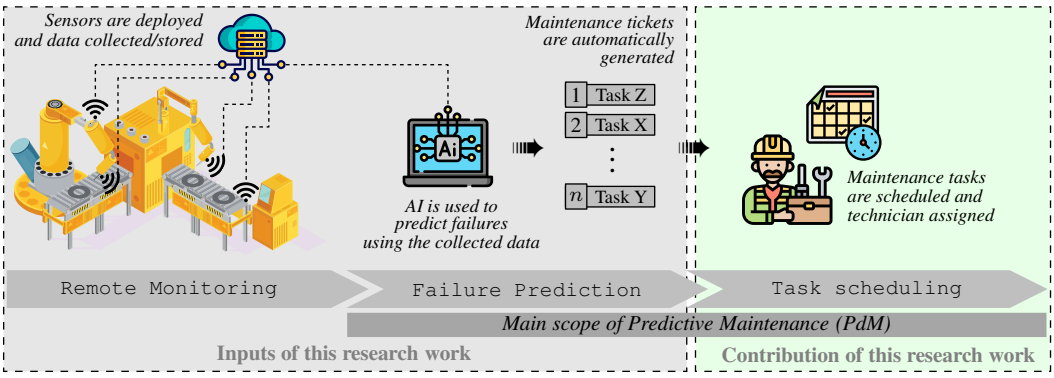


Figure 2.1: Key stages to move towards PdM

Having an effective maintenance provides a reliable production which converts to a profit as the final goal of any company. Maintenance involves two stages.

- **Maintenance Planning**: It is the process of gathering all the requirements, including the identification of the possible tools and parts to be used to perform the maintenance, the identification of the technicians with their skills and availability, and the procedure to be performed for the maintenance.
- **Maintenance Scheduling**: It is the process of assigning the different maintenance tasks to the technicians based on their skills, availability and priority of interventions. In this stage, an initial schedule of activities to be carried out is established, which can be affected or modified by new interventions, changes in priorities or unexpected delays.

Maintenance activities may involve different actions, such as inspection, where the equipment is examined to identify possible irregularities; repair, where the condition of the equipment is restored to a previous or original state; and replacement, where the machine or some component of the machine are replaced by another one in a better state (ideally a new component).

There are different maintenance strategies that can be implemented [AKA24]:

- **Corrective Maintenance (CM)**: Is the simplest maintenance policy, which only performs maintenance actions once the equipment suffers a breakdown. The advantage of this type of maintenance policy is that it allows the full use of the equipment's useful life. However, sudden failures can affect the scheduling of production and maintenance resources.
- **Preventive Maintenance (PM)**: It is a proactive policy where the objective is to schedule maintenance activities on a regular basis in order to reduce unplanned breakdowns. However, these maintenance activities are often performed unnecessarily, which can increase operational costs.
- **Reliability-Centered Maintenance**: It is a systematic approach that establishes a baseline for the equipment, identifies potential failures and their consequences, and develops appropriate maintenance tasks to enhance reliability, optimize costs, and improve overall operational efficiency.
- **Total Productive Maintenance**: Is a hollistic approach involving all employees (e.g. operators, technicians, managers) in the maintenance activities to maximize productivity by ensuring all equipment are in optimal conditions.
- **Predictive Maintenance (PdM)**: This is the most advanced maintenance policy, which aims to monitor equipment with sensors and predict when a machine is likely to fail. The objective is to schedule the maintenance intervention just before the failure occurs, thus maximizing the useful life of the equipment and ensuring optimal performance in the maintenance operations.

An increasing number of PdM systems are emerging in all sectors of the industry [LHY+20]. Some market studies report that PdM can reduce time required to plan maintenance by 50%, increase equipment uptime by 20% and reduce costs by 10% [CDC+22]. A PdM system can be seen as a three stage-process, as depicted in Figure 2.1:

1. *Remote monitoring:* sensors and cameras are deployed on production lines to capture real-time events and communicate these data to on site or cloud systems;

2. *Failure Prediction:* analytics tools with predictive (ML) capabilities analyze the collected data to determine when a system (component, equipment, process) is likely to fail;

3. *Task Scheduling:* based on the computed predictions, and expert knowledge, decision support systems are designed, which often include the optimization of maintenance task scheduling and operator assignment.

Each of these stages is the subject of significant research efforts, for example to address interoperability issues in stage (1) [TQL+18], to make AI (Machine

Learning - ML) models more robust, accurate and transparent for predicting failures in stage (2) [CZS+22; YRZ+22], or still to improve state-of-the-art maintenance scheduling strategies [LDC18] and better model expert knowledge in the decision process [BLA+19; GP21].

As emphasized in Figure 2.1, although stage (1) (remote monitoring) is a prerequisite for PdM, the main goal of any PdM framework is to predict and optimally schedule the maintenance interventions/tasks.

2.2 Reinforcement Learning

Reinforcement Learning (RL) is a subset of machine learning in which an agent (decision maker) must learn a behavior (called a policy) through trial and error in a dynamic environment [SB18]. In this sense, the agent is connected to the environment through its perception and actions, as illustrated in Figure 2.2.

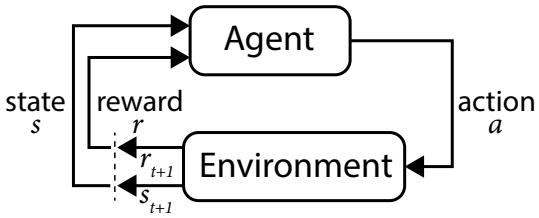


Figure 2.2: RL framework diagram

During each interaction with the environment, the agent receives an observation o from the environment. This observation is a partial description of the state of the environment s. If the agent is able to observe the complete information from the environment, it its called fully observable, otherwise the environment is partially observable. The set of actions that the agent is able to take forms part of the action space. These actions can have discrete or continuous values. In order to decide which action a the agent should take based on an observation, the agent can follow a policy π. This policy can be deterministic, whereby for each observation the agent will always perform the same action, or stochastic, whereby every action has a probability of being executed. A policy can be defined as

$$a \sim \pi(s)$$

When performing the action on the environment, the agent receives a value indicating how well or poorly the transition through the new state is. This value is called the reward r and is obtained trough the reward function defined as

$$r = R(s, a)$$

The objective of the agent is to maximize the cumulative reward, referred to as the return, G, over the entire horizon, H. This can be expressed as follows:

$$G = \sum_{t=0}^{H} \gamma^t r$$

The term γ is used to reflect the importance of immediate or future rewards, particularly in some RL problems where there is a long-term or infinite horizon. In this context, the discount factor γ (with $0 \leq \gamma \leq 1$) reflects how myopic is the agent to the future rewards.

For the agent, it is important to to know what is the expected return starting from a state or state-action. That is, following a policy π the state-value function $v_\pi(s)$ and action-value function $q_\pi(s, a)$ are defined as:

$$v_\pi(s) = \mathbb{E}[G \mid s]$$
$$q_\pi(s, a) = \mathbb{E}[G \mid s, a]$$

Ideally, the problem of RL is to find the optimal policy represented by $\pi^*(s)$ in which the action a maximizes the expected return from starting in s.

RL algorithms can be classified in two key ways [Mor20]. The first classification is based on how much knowledge they have of the environment.

- **Model-based algorithms** are the ones who understand completely how the environment works and its dynamics. For this, the algorithm understand how the transition goes from one state to the other and what reward to expect from those transitions.
- **Model-free algorithms** are usually implemented when the dynamics of the environment is too complex to model and the agent learns from the interactions with the environment.

The second classification is about what exactly is the algorithm learning and how to translate it to the policy.

- **Value-based methods**: Refers to algorithms that learn value functions which estimates the expected return (cumulative reward). The policy is derived from the value function. Some examples are Q-learning[SB18], SARSA[SB18], DQN[MKS+13].
- **Policy-based methods**: Refers to algorithms that optimize directly the parameterized policy without explicitly learning the value function. Some examples are REINFORCE [SB18], TRPO [Sch15]
- **Actor-Critic methods**: Refers to methods that learn both a policy and a value function, which is responsible for selecting actions, and the critic, which

evaluates the actions by estimating value functions. Some examples of this algorithms are PPO [SWD+17], A3C [Mni16]

During this thesis we focused on model-free RL algorithms, in particular the use of two RL algorithms, PPO and DQN, which will be described in the following chapters.

3

Related Work

This chapter reviews existing work related to maintenance optimization with a focus on uncertainty and sustainability.

Contents

3.1 Uncertanty in maintenance scheduling

3.1.1 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 [MDZ+22] who propose a modified non-dominated neighbor immune algorithm for the allocation of technicians and spare parts, the objective being twofold: reducing both tardiness and maintenance cost. The authors compare their approach against other multiobjective algorithms under different conditions (i.e., varying the number of machines, technicians, and maintenance tasks), whose results show that it performs better in most cases. [YAK+22] 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 [WBF+20] have used ML (clustering) techniques such as k-means, mean-shift, Expectation-Maximization, or Density-Based Spatial Clustering of Applications with Noise 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, [NKT+20] 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. [SYP19] 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.

3.1.2 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 [GDA+23]. In this respect, [CD20] 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 CBM policies. The results show a significant improvement in the profit gain. [DNP+19] 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 implemented different heuristics and made use of CPLEX with randomly generated instances. [RSL20] 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. [LCZ21] also use a Bayesian approach to predict the 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 seat and is compared with a classical maintenance policy (considerable savings are obtained). [AR20] 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 PM to minimize the total time between PM actions and CM. Artificial data is used to exhaustively vary different parameters in order to demonstrate the results of parameter selection for failure prevention. [ADI+22] 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 that establishes different costs for maintenance and production operations.

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 [RF21] 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 savings in computational power. [BLD22] also developed a multi-agent 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. [KJL19] 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 CM and PM policies with respect to completed jobs, while evidencing how agents learn to execute maintenance closer to failure times with a low buffer volume. [VAW+22] also tackle the problem of opportunistic maintenance using RL in a wafer manufacturing plant scenario. The proposed approach is compared with DR-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 [YWW22] who addresses the problem of flexible job-shop problem integrated with time-based maintenance and CBM. An RL algorithm called "double-layer Q-learning" is designed to select machines and jobs in a dynamic way. This work was evaluated against several metaheuristics, which showed the relevance of the approach.

3.2 Economic-driven maintenance scheduling

From the three pillars of sustainability, it can be observed that the majority of articles focus on the economic aspect, as they are primarily business-oriented. As illustrated in Figure 6.1, there is a clear preference for addressing the optimization of maintenance through metaheuristic methods. Evolutionary computation is one of the preferred methods for optimizing maintenance. For example, [SFZ20] addresses the problem of two-stage scheduling in an assembly flow shop with random machine breakdowns. In this work, the objective is to minimize the expected weighted sum of the makespan and the mean completion time. Maintenance is performed as part of the completion time for each processing time of each job on the machines. Four metaheuristic algorithms are proposed to solve this optimization problem: GA,

cloud theory-based Simulated Annealing (SA), Imperialist Competitive Algorithm, and New Self-adapted Differential Evolutionary. The last approach showed to be statistically superior to the other proposed meta-heuristic algorithms in terms of solution quality and computational time. Similarly, in their study, [WDZ+23] addressed the challenge of joint optimization of integrated mixed maintenance and distributed two-stage hybrid flow-shop production for multi-site maintenance requirements. Their objective was to minimize the total weighted earliness/tardiness penalty and the number of lost orders. To achieve this, they proposed an Improved Non-dominated Sorting Genetic Algorithm II (NSGA-II), which demonstrated superior performance compared to other algorithms utilized in the study. Furthermore, [BB19] addressed the problem for a joint production and maintenance policy under non-renewable resource constraints (e.g. raw materials or fuel). Two objectives were defined, to minimize the expected makespan and the total maintenance cost for preventive and corrective actions. NSGA-II was adopted to solve the optimization problem. A numerical examples were generated to evaluate the proposed solution against LINGO 10. The results presented different trade-offs of the solutions for the multiobjective problem. In addition, [CGG21] worked on a multi-objective optimization for PM policy, where different experiments were conducted on the NSGA-II to optimize system availability (maximize) and operating cost (minimized). The approach was evaluated in a case of study in which optimization of the design and PM strategy needed to be implemented for an industrial fluid injection system. Other metaheuristic methods have been implemented to solve the maintenance problem as presented in [BHK21], where the objective is to minimize the makespan for task scheduling in identical and parallel machines. For this problem, a heuristic approach was implemented, together with an implementation of Tabu Search (TS). The results presented show insights about the different performance of the methods evaluated, which are dependent on the shape of the intree that is formed by the constraints of the ordered tasks. Other works, as investigated by [QAA+20] addressed the problem of a joint production and maintenance policy for a single machine environment with the objective of minimizing the number of tardy jobs. In this work a Mixed-Integer Linear Programming model and an ACO model were proposed for optimizing small and large sizes instances, respectively. In addition, Moore's algorithm was used to evaluate ACO solutions, showing how ACO outperform Moore's algrithm for all the instances tested. [BM21] worked on the problem of joint production and maintenance policy with the aim to optimize mean tardiness, schedule instability, makespan and mean flow time. To solve this, a Greedy Randomized Adaptive Search Procedure algorithm is tested. The approach was evaluated against a set of DR and a SA algorithm, showing the advantage of the proposed method as an effective approach for improving the performance of dynamic flexible job shop scheduling problem. The work by [HAG+21] address

the problem of joint production and maintenance with parallel machines with a single server and unavailability constraints with the objective to minimize the makespan. To solve this problem, a lower bound and three metaheuristics (SA, TS and GA) were proposed where a different tradeoff of solutions for different subset of instances were showed. Additionaly, [DDC+19] worked on a joint production and maintenance policy for identical parallel machines with the objective of minimizing the making space of all activities. The model was solved in a hybrid way, combining an approach based on DR and SA. The approach was evaluated in a case study on a plastic industry with seven identical injection machines and different alternative methods based on different rules and SA.

In contrast, other works have chosen to employ classical methods such as MP. [FTS19] explored a mixed integer non-linear programming model to determine the optimal PM interval represented by a continuous-time Markov chain. The objective is to reduce costs per unit time considering both perfect and imperfect maintenance levels. The model was evaluated in a numerical example and solved by the Baron solver. [DNP+19] worked on the problem of joint production maintenance scheduling where the objectives are the completion time and makespan considering different robust criteria. Maintenance activities are considered and are seen as additional tasks. The problem was solved using mixed integer linear programming and a heuristic method tested on randomly generated instances.

A limited number of studies have employed AI/ML techniques. [JAA21] worked on a maintenance management system for tire manufacturing that provides an optimum time frame for PM actions on condition monitoring and production data. The scheduler was implemented using an artificial neural network where condition monitoring data, sensor inputs, and machine operator inputs are used to generate breakdown alarms, dynamic scheduling, and breakdown alarms. The system was evaluated using performance charts and regression using real data from the manufacturing plant. [GLG+21] worked on the problem of maintenance scheduling in a production environment where maintenance actions are decided by a policy trained using RL. The evaluation was tested on a discrete event simulator where machines suffer from different degradation states, a buffer level is considered, and maintenance actions are performed by the decision maker (the agent). The objective is to minimize the maintenance cost and maximize the cumulative value of the products.

Other agent-based systems have also been used to tackle this kind of problem. [HLB19] worked on maintenance scheduling on geodistributed assets in a distributed industrial environment. In this work, a multi-agent system solution was implemented where the mechanism promotes competition and cooperation of agents to obtain a global schedule. The results performed well in terms of Global Cost, Total Weighted Tardiness Cost, and makespan compared to Weighted

Shortest Processing Time first–Heuristic–Earliest Due Date method. [BLD22] worked on the problem of scheduling maintenance activities for a joint production and maintenance policy in a multi-agent approach. The proposed approach modeled the problem as a multiagent with the objective to solve several subproblems (as customers, producers, managers) where all these agents are coordinated by a supervisor agent. The framework is evaluated based on different key performance indicators (KPI) such as execution time, number of cycles, number of maintenance tasks, number of late jobs, total tardiness, and load of each machine. In a similar work, [KPN+24] worked on joint maintenance and production scheduling with the objective of optimizing maintenance cost and production-related metrics. In their work a multi-agent system is proposed where they improve a Contract-Net protocol by addressing the challenges of agent myopia leading to suboptimal resource allocation and performance losses. Their work was tested in a simulation environment that showed improvements in terms of scheduling efficiency, maintenance cost, and execution times.

Other methodologies have been employed to address the economic aspects of maintenance optimization. The study by [DS19] examines the effect of different criticality policies based on the shortest mean time between failures, longest queue length, high level of utilization, and longest mean repair time. These policies were evaluated with a discrete-event simulation model that represents the dynamic of a real-world manufacturing cell that includes machine failure, maintenant resource allocation, material flow, job sequencing and scheduling. Policies were evaluated on the basis of machine availability and mean throughput time. [RF22] worked on the maintenance scheduling problem where the objective is to maximize revenue and availability (to find the optimal prices of goods and optimal maintenance scheduling); for this the authors proposed a two-level optimization solution based on game theory called leader-multiple-followers game. In this, customers are considered as followers seeking to obtain their consumption and the supplier as the leader who is responsible for obtaining the price of the network and maintenance scheduling of its manufacturing units, demonstrating the effectiveness of integrating network externalities and predictive maintenance scheduling in pricing strategies for suppliers, leading to increased revenue and profit. [Cha23] worked on the problem to determine the best schedule for a preventive replacement last policy with the objective of minimizing the mean cost rate over a finite time horizon. A numerical example was presented in order to minimize the total mean cost rate incorporating costs related to repairs, maintenances, replacements, inventory and shortage. In a study by [Al-20], a Criticality Analysis was conducted for a pharmaceutical company where a risk matrix was used to classify equipment and set priority levels. Historical maintenance data was used to evaluate the current status of the machine. The analysis obtained showed that the implementation of the rescheduled PM

improve the maintenance effectivenes, reducing failures and optimizing the resource utilization. [YH21] studied the problem of machine scheduling focusing on periodic machine maintenance for single-machine and flow shop scheduling models. The objectives were to minimize the total completion time and minimize the maximum lateness. Random examples were generated for the model studied and solved with a proposed algorithm called Smallest Sum of Processing and Removal Time First and improved with a Minimum Cost Insertion. [SZ19] worked on the problem of maintenance scheduling for a parallel machine setup. In this problem, processing and maintenance time are considered uncertain variables. To solve this problem, an improved version of a Long Processing Time rule is implemented, and it is tested with numerical experiments and compared with a Heuristic Method called HPSOGA that is a combination of GA and Particle Swarm Optimization. The results showed the performance of the improvement version against the original rule, and narrowing respect to the HPSOGA. Simulation-based approaches have been also used to address the maintenance optimization as presented by [WYD+20]. In this work, they tackle the problem of joint production and maintenance policy for failure-prone parallel machines in make-to-order production environment. The objective of the model is to minimize the weighted long-run average waiting costs of the production system. A simulations were performed using the Value Iteration algorithm and compared against different DR in the literature demostrating an improvement in the jobs waiting time and average machine downtime. Furthermore, [GJ22] worked on a stochastic flexible job shop scheduling that considers uncertainties and dynamic jobs arrivals. In this work, five input parameters are considered, that is, reliability-centered PM, percentage of machine failure, mean time to repair for random machine breakdown, due date tightness factor, and routing flexibility. Additionally, [GJC+23] continue working on the problem of flexible job shop scheduling under reliability-based PM. In this work, a simulation-optimisation approach was implemented in which the mean flow time, the maximum flow time, the mean tardiness and the number of late jobs were evaluated in different scenarios. Finally, [LRC+23] worked on the problem of opportunistic maintenance for a multi-unit system using Monte Carlo simulation. The main objective was to optimise the maintenance cost where it is shown using simulated data to model the performance of a Computer Numerical Control (CNC) machine.

3.3 Towards sustainable maintenance scheduling

3.3.1 Sustainable scheduling: energy consumption as prime focus

For sustainable maintenance policies, environmental aspects are usually addressed in terms of reducing the energy consumption. As was evidenced in Fig-

ure 3.1c, most articles make use of metaheuristic methods to solve sustainable maintenance scheduling. [SDL20] propose a joint energy, maintenance, and production model with the objective of minimizing the total electricity cost, maintenance cost, and minimizing production loss based on the production throughput of the manufacturing system. To evaluate the model, a numerical study with five machines and four buffers was implemented where particle swarm optimization was used to solve the model. [WDX+20] also consider maintenance and peak power consumption for the problem of collaborative optimization of manufacturing scheduling for the hybrid flow shop. The objective is to minimize the makespan considering maintenance plans and peak power consumption. Despite these two research works, most studies use evolutionary algorithms. [GTS+20] studied a joint production and maintenance policy in a single machine considering machine deterioration and failures. The goal is to minimize the total cost including inspection, repair, energy consumption, and the makespan. The model is solved using GA and is evaluated against SA and imperialist competitive algorithms in artificial instances. Similarly, [ST21] worked on a single machine level for the joint production and maintenance schedule with multiple failures, inspired by a real-world problem for a manufacturing company on a Lathe CNC machine. The objective is to optimize the total cost of the system (incl. maintenance cost, machine energy consumption cost, makespan). In this work, GA, SA, and teaching-learning-based optimization algorithms are implemented, with GA demonstrating superior performance. [SC20] propose a bi-objective optimization model for a single machine considering electricity cost and PM using GA. The objectives are to minimize the total energy cost and machine unavailability. The results show that the proposed hybrid multiobjective GA yields better outcomes than the NSGA-II and is faster than the Baron solver. The work of Mirahmadi and Taghipour [MT19] goes beyond the single machine level to the flexible job-shop scheduling problem considering maintenance, production, and energy aspects with the objective of minimizing the expected makespan. The optimization model is solved using GA and tested on a small scale for three industrial machines and four jobs. [ACZ+20] optimize – *using a multi-objective evolutionary algorithm with the pareto elite storage strategy* – the production process considering maintenance by minimizing the makespan, total tardiness, total production cost, and total energy consumption. Cui et al. [CL20; CSX20; CL21] also work on the problem of joint production and maintenance optimization involving energy consumption, which is also solved by GA.

Other studies employ alternative methods beyond metaheuristics. [ASjK19] develop a joint production-maintenance policy to optimize – *using the Kuhn-Tucker method* – production quantity, production rate, and manufacturing reliability considering variable energy consumption cost. The study presents several insights, including that a controllable production rate is preferable when dealing with an

unreliable manufacturing system and that the expected total cost is influenced by decision variables such as production quantity, production rate, and reliability parameter. [XSS+21] develop a joint policy that considers PM and replacement policies in an energy efficient way. The aim is to minimize the total non-value-added energy consumption obtaining the best intervals for preventive and replacement maintenance. The model is evaluated with data collected from Boehringer NG200 Crankshaft Turning CNC. Furthermore, [XSS+22] studied an energy-oriented selective maintenance policy for series-parallel multi-unit systems, the objective being to maximize energy efficiency by optimizing the maintenance actions of machines at each breakdown. The model is solved using a modified branch-and-bound algorithm and tested on a numerical example based on the production system of an engine craft. Lastly, [GJ21] study reliability-based maintenance in a simulation environment considering setup time and energy-related metrics. The approach is evaluated against makespan, mean flow time, mean tardiness, number of tardy jobs, total setup time, average operating energy consumption, and average idle energy consumption. The results demonstrate the advantage of a reliability-centered periodic PM approach.

3.3.2 Beyond energy consumption

In addition to energy consumption, reducing the overall system carbon footprint (e.g., a production line, process) is one of the main objectives of today's companies. [AAA+24] studied the relationship between the production makespan, maintenance activities, energy consumption, and carbon footprint (maintenance activities are of the highest importance in this relationship). [DY20] address a two-stage joint optimization problem of green manufacturing and maintenance for semiconductor wafers considering the inspection and repair stages simultaneously. The authors present a hybrid multi-objective multiverse optimization algorithm that minimizes the makespan, total carbon emissions, and total PM cost. Finally, [MFZ+20] address the problem of maintenance scheduling for complex equipment considering green performance, which is divided into two sub-objectives: (i) minimizing the global maintenance cost resulting from resource consumption, production delay, and fault risk; (ii) minimizing carbon emissions during maintenance resource scheduling. The algorithm was implemented using NSGA-II and tested in a use case involving a grinding roll fault in a large vertical mill. Another study [PKK23] focuses on the environmental pillar by examining different aspects of manufacturing and waste management. An optimization framework for process planning in a degrading multi-state system is proposed in this respect. The objective is to increase revenues through the sale and production of high-quality products, recycling, and minimizing production and maintenance costs.

In addition to the environmental pillar, we found a unique article by [QZL+22] that addresses the social dimension. This study aims to minimize the number of

man-days required for temporary employment in the maintenance management of the steel industry. This goal is achieved by reducing the number of maintenance days and minimizing the deviation between the interval times of maintenance tasks and the maintenance period of the equipment nodes. To this end, a two-stage optimization strategy is developed, which consists of generating a pre-schedule using a rule-based method, followed by a GA-based optimization. The test with real data from a Baowu steel company showed an improvement of 40.3% on the basis of pre-scheduling.

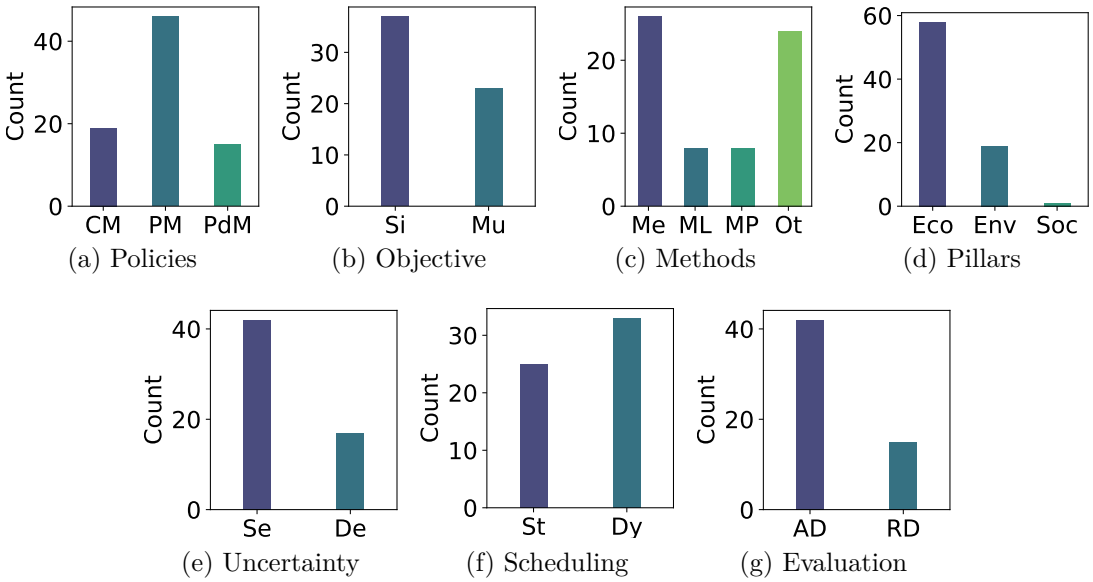
3.4 Summary

This section provides an overview of the state-of-the-art approaches used for maintenance scheduling in manufacturing based on six criteria, each one taking two or more values, as detailed hereinafter:

- **Maintenance Policy:** Corrective Maintenance (CM), Preventive Maintenance (PM), Predictive Maintenance (PdM)
- **Type of objective:** Single (Si), Multiple (Mu)
- **Solution method:** Metaheuristic (Me), Machine Learning (ML), Mathematical Programming (MP), Others (O).
- **Sustainability Pilars:** Economic (Eco), Environmental (Env), Social (Soc)
- **Uncertainty:** Stochastic (Se), Deterministic (De)
- **Scheduling:** Static (St), Dynamic (Dy)
- **Evaluation:** Real Dataset (RD), Artificial Dataset (AD)

To ease the analysis, Figure 3.1 offers an overview of the distribution of the reviewed articles by criterion.

Most of the maintenance policies developed in the articles are PM (see Figure 3.1a). Although PdM has a higher level of maturity compared to CM, the last one is still more commonly used. In addition, Si problems are addressed slightly more often than Mu problems (see Figure 3.1b). Regarding the optimization of maintenance scheduling, Me methods represent the standard solution approach (see Figure 3.1c) with a lack of research on ML/MP approaches. On the sustainability pillars (see Figure 3.1d), the majority focus on the Eco pillar, as they are primarily business-oriented. Interestingly, we found that for the different types of uncertainty (see Figure 3.1e), most of the works address a Se environment that cover a widely types of uncertainty even though the PdM policies are not highly address. We also found that most of the articles that address different types of uncertainty work with a Dy scheduling with St scheduling slightly below (see Figure 3.1f). Finally, most articles use AD for evaluation purposes, as shown in Figure 3.1g, with few real-use cases (RD). Additionally, Table 3.1 consolidates these findings, offering a detail and concise overview of the trends observed in the literature.



(a) Policies (b) Objective (c) Methods (d) Pillars

(e) Uncertainty (f) Scheduling (g) Evaluation

Figure 3.1: Distribution of research articles per category

Table 3.1: Summary table of the literature review

Article	Policy	Obj.	Method	Sust.	Env.	Sched.	Eval.
[Al-20]	PM	Si	Ot	Eco	Se	Dy	RD
[AR20]	CM,PM,PdM	Si	Ot	Eco	Se	Dy	Ad
[ACZ+20]	PM	Mu	Me	Eco, Env	De	St	-
[ADI+22]	PdM	Si	Ot	Eco	Se	Dy	Ad
[ASjK19]	CM	Si	Ot	Eco, Env	Se	Dy	AD
[AAA+24]	CM, PM	Si	MP	Eco, Env	Se	St	AD
[BM21]	PM	Mu	Me, Ot	Eco	Se	Dy	AD, RD
[BHK21]	PM	Si	Me	Eco	De	St	AD
[BLD22]	PdM	Mu	Ot	Eco	Se	Dy	AD
[BB19]	CM, PM	Mu	Me	Eco	Se	St	-
[CGG21]	PM, PdM	Mu	Me	Eco	Se	Dy	RD
[CD20]	PM,PdM	Si	Me	Eco	Se	Dy	Ad
[Cha23]	PM	Si	Ot	Eco	Se	St	AD
[CL20]	PM	Mu	Me	Eco, Env	De	St	AD
[CSX20]	PM	Mu	Me, MP	Eco, Env	De	St	AD
[CL21]	PM	Mu	Me, MP	Eco, Env	De	St	AD

Article	Policy	Obj.	Method	Sust.	Env.	Sched.	Eval.
[DNP+19]	PM	Mu	MP, Ot	Eco	Se	Dy	AD
[DDC+19]	PM	Si	Me, Ot	Eco	Se	St	RD
[DS19]	CM, PM	Mu	Ot	Eco	Se	Dy	AD
[DY20]	PM	Mu	Me	Eco, Env	Se	Dy	AD
[FTS19]	PM	Si	MP	Eco	Se	Dy	AD
[GTS+20]	PdM	Si	Me	Eco, Env	Se	Dy	AD
[GLG+21]	PdM	Si	ML	Eco	Se	Dy	AD
[GJ21]	PM, PdM	Mu	Ot	Eco, Env	Se	Dy	AD
[GJ22]	PM	Mu	Ot	Eco, Env	Se	Dy	AD
[GJC+23]	PM	Si, Mu	Ot	Eco	Se	Dy	AD
[HLB19]	CM	Mu	Ot	Eco	Se	Dy	AD
[HAG+21]	PM	Si	Me, Ot	Eco	De	St	AD
[JAA21]	PM, PdM	Mu	ML	Eco	De	Dy	RD
[KPN+24]	PM	Mu	Ot	Eco	Se	Dy	AD
[KJL19]	PdM	Si	ML	Eco	Se	Dy	Ad
[LRC+23]	CM, PM	Si	Ot	Eco	Se	St	RD
[LCZ21]	PdM	Si	N/A	Eco	Se	Dy	Rd
[MFZ+20]	CM, PM, PdM	Si	Me	Eco, Env	Se	Dy	AD, RD
[MDZ+22]	CM	Mu	Me	Eco	De	St	Ad
[MT19]	CM, PM	Si	Me	Eco, Env	Se	St	AD
[NKT+20]	CM,PM,PdM	Si	Me	Eco	Se	St	N/A
[PKK23]	CM, PM	Si	ML	Eco, Env	Se	Dy	AD
[QAA+20]	PM	Si	Me, MP	Eco	De	St	AD
[QZL+22]	CM, PM	Si	Me, Ot	Eco, Soc	De	St	RD
[RF21]	PdM	Si	MP	Eco	Se	Dy	Rd
[RF22]	PdM	Mu	Ot	Eco	Se	St	AD
[RSL20]	PM	Mu	ML	Eco	Se	Dy	Rd
[SFZ20]	CM	Si	Me	Eco	Se	Dy	AD
[ST21]	CM, PM	Si	Me	Eco, Env	Se	Dy	AD
[SZ19]	PM	Mu	Ot	Eco	Se	St	AD
[SC20]	PM	Mu	Me	Eco, Env	De	St	AD
[SYP19]	PM	Si	Ot	Eco	Se	St	Rd
[SDL20]	CM	Si	Me	Eco, Env	De	St	AD
[VAW+22]	CM,PM	Si	ML	Eco	Se	Dy	Rd
[WDX+20]	PM	Si	Me	Eco, Env	De	St	AD
[WDZ+23]	CM, PM	Mu	Me	Eco	De	St	-
[WBF+20]	PM	Si	ML	Eco	De	St	Rd
[WYD+20]	PM	Si	Ot	Eco	Se	Dy	AD
[XSS+21]	PM	Si	Ot	Eco, Env	Se	Dy	RD
[XSS+22]	PM	Si	MP	Env	Se	Dy	AD
[YWW22]	CM,PM	Si	ML	Eco	Se	Dy	Ad
[YAK+22]	PM	Si	Me	Eco	De	-	AD
[YH21]	PM	Si	Ot	Eco	De	St	AD

4

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

The scheduling of maintenance processes can be affected by different types and degrees of uncertainty, which makes it crucial to analyze their impact on the effectiveness of various optimization techniques. In this chapter, we explore how different optimization approaches based on dispatching rules, genetic algorithms and reinforcement learning respond to these uncertainties in maintenance planning in an industrial environment.

Contents

4.1 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 [GDA+23]. The maintenance process works to prevent and correct failures by restoring or replacing the components that caused the machine to fail.

Figure 4.1 illustrates a possible processing flow to schedule maintenance tasks, starting from the collection of sensor data to estimate the 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.*) [DS20]. Despite the prevalence of mathematical programming [SSS+23; Sto23] and metaheuristic methods to solve maintenance scheduling problems, uncertainty remains a challenge [GZT+23; ZTC+23] 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 [DKT+15]. Uncertainty can come from different sources such as: estimation of RUL of equipment [BMM+20], duration of maintenance time [YLC16], availability and efficiency / fatigue of the technician [FAP+17], or the occurrence of unexpected events such as sudden failures and unavailability of the technician.

To address this dynamic and stochastic problem, methods such as hybrid simulation optimization (also known as Simheuristic) [AR20], ML [SHA+22], and rule-based are often employed. Among these, GA is frequently utilized as part of these methods for hybrid simulation optimization. This is a search and optimization algorithm inspired by the evolutionary principle, taking advantage of its capability to efficiently explore and optimize complex solution spaces. It is widely used to solve optimization problems, such as exergoeconomic optimization for geothermal power plants [NNA+18], stock market prediction [DZY+24], PID optimization [ZAO+20], communication networks [NGY+23] among others. On the other hand, DRL has been applied across various domains to obtain optimal (or near-optimal) policies, even in environments with varying complexity and uncertainty. Applications range from vehicle routing [PL23], drone racing [KBL+23], protein design [LWN+23], and even sectors with high-risk implications such as healthcare care [YSW+23]. Although these methods are widely used to optimize maintenance scheduling,

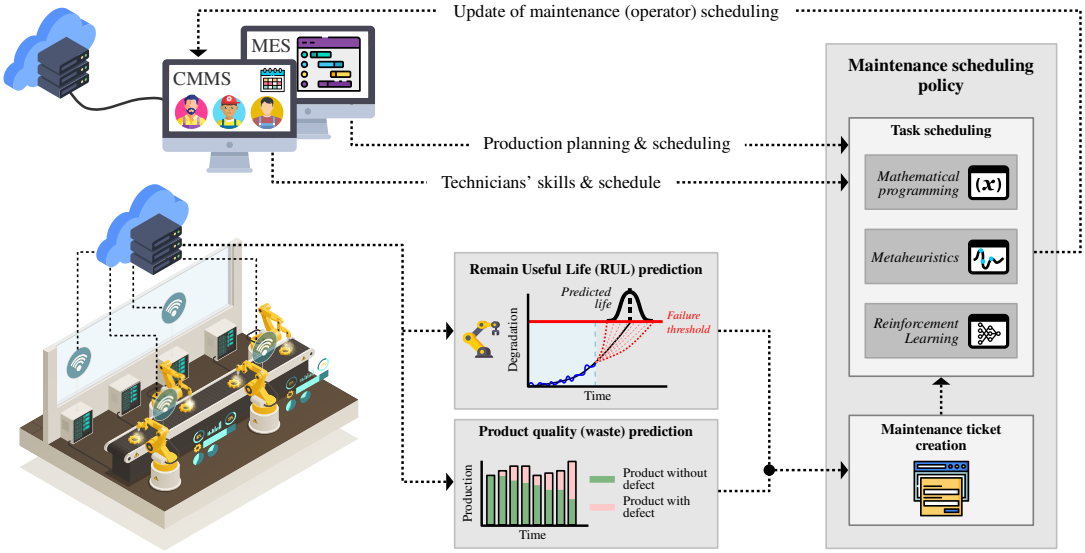


Figure 4.1: Workflow diagram of the automated maintenance scheduling system

there is still a lack of understanding of how they behave under different kinds of uncertainties. To fill this gap in the literature, this chapter aims to explore the performance of GA, DRL, 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 4.2 further discusses the maintenance scheduling problem, along with state-of-the-art methodologies and methods to solve it. Section 4.3 formally describes the maintenance scheduling problem and the optimization methods considered for benchmarking purposes (incl., DRL, GA-Simheuristic, DR). Section 4.4 presents the results obtained considering both simulated maintenance scenarios and a real-life use case; discussion and conclusions follow.

4.2 Scheduling under uncertainty

4.2.1 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 [DS20]:

- **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 [YZC+19];

- **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 [FAP+17];
- **Technician availability:** technician might unexpectedly request sick leave or vacation, or may need to be reassigned to a task of higher priority [RKdG+22];
- **Machine availability:** unexpected failures may occur at any time, or a change of priority in the production schedule [HCA20];
- **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 [SBS+21];
- **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 [WBF+20];

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

4.3 Maintenance scheduling approaches

To evaluate and compare different optimization methods under different degrees of uncertainty, a two-stage approach is adopted, as illustrated in Figure 4.2. The first step consists of building the environment to simulate the maintenance process in a job shop floor, where both machines and technicians are modeled (*cf.*, Figure 4.2). The uncertainty associated with machines and technicians, respectively, relate to failure distribution and TTR. The second stage consists of the definition of the three optimization methods (DRL, GA-S, DR) and the way to compare them. These two stages are further detailed in sections 4.3.1 and 4.3.2 respectively.

4.3.1 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.

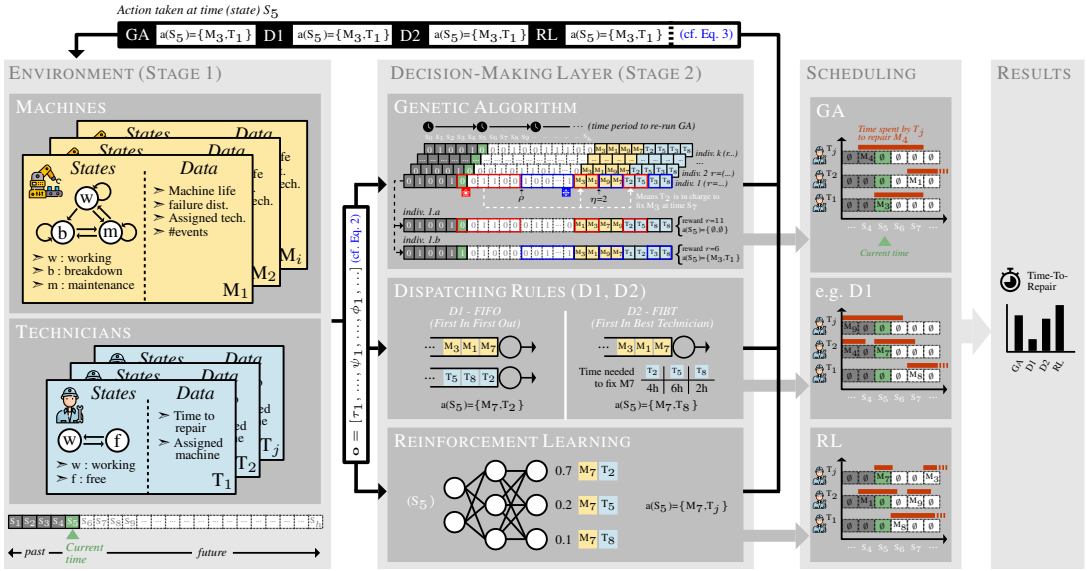


Figure 4.2: Overview of the two-stage dynamic maintenance scheduling framework

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 TTR 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 $\sum_{j \in \mathcal{J}} z_{ijt} = \{0, 1\}$. 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 (4.1). Table 4.1 summarizes all the variables used in this paper.

$$\max \sum_{\forall t \in H} \sum_{\forall i \in \mathcal{M}} \mathbb{I}(x_{it} = 0) \tag{4.1}$$

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 $x_{it} = 0$ throughout the time horizon H.

Table 4.1: 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
$\mathcal{D}$	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}	Remainig maintenace time of technician i to machine j at time t
μ_{ij}	Mean of the TTR for machine i by technician j
σ_{ij}	Standard deviation of the TTR 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 remainingn 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

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 TTR 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 $\mathcal{P} : S \times A \to S$: 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 $R : S \times A \times S \to \mathbb{R}$: 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 $\gamma \to 1$ 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:

$$\mathbf{o} = [\tau_1, ..., \tau_{|\mathcal{M}|}, \psi_1, ..., \psi_{|\mathcal{M}|}, \phi_1, ..., \phi_{|\mathcal{T}|}] \tag{4.2}$$

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:

$$\mathbf{a} \in \mathcal{M} \times \mathcal{T} \cup \{d\} \tag{4.3}$$

The equation (4.3) 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:

$$r(\mathbf{o}, \mathbf{a}, \mathbf{o}') = \frac{\sum_{i \in \mathcal{M}} (1 - \frac{\eta_{ijt}}{\max d_{ij}})}{|\mathcal{M}|} \tag{4.4}$$

Where each of the components of (4.4) 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 efficency, 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 substraction 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 TTR; 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.

4.3.2 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 4.3.2 to 4.3.2, respectively, introduce the proposed DRL, GA-simheuristic (GA-S), and DR to solve the dynamic maintenance scheduling problem formalized in Section 4.3.1.

Reinforcement learning

During exploration, the DRL agent has to select a technician and a 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 nonviable solutions. In the present paper, we propose to use the Proximal Policy Optimization (PPO) [SWD+17] 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 (4.5)-(4.6), 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 of the softmax function.

$$mask(l(s))_i = \begin{cases} l_i & \text{if } a_i \text{ is valid in } s \\ M & \text{Otherwise} \end{cases} \tag{4.5}$$

$$\pi_\theta^{'}(\cdot|s) = \text{softmax}(mask(l(s))) \tag{4.6}$$

GA-S

GA is a popular alternative for the static scheduling problem [NKT+20], 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-S, the encoding of the chromosome, at time t, is a vector of the form $[\mathbf{e}, \mathbf{b}, \mathbf{c}]$, where:

$$\mathbf{e} = [e_0, ..., e_H] \qquad e_i \in \{0, 1\} \tag{4.7}$$

$$\mathbf{b} = [b_0, ..., b_{|\mathcal{M}-1|}] \qquad b_i \in \mathcal{T} \tag{4.8}$$

$$\mathbf{c} = [c_0, ..., c_{|\mathcal{M}-1|}] \qquad c_i \in \mathbb{Z} \tag{4.9}$$

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 $[e_0^1 : e_{\rho-1}^1]$ of the child $\mathbf{e^1}$ (as shown in Figure 4.2 through ❀ symbol) will be permuted and the elements of $[e_\rho^2 : e_H^2]$ of the child $\mathbf{e^2}$ (as shown in Figure 4.2 through ❇ symbol) will be permuted. Let us consider $\eta = \sum_{i=0}^{\rho} e_i^1$, 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 $[b_0^1 : b_{\eta-1}^1]$ and $[b_\eta^2 : b_{|\mathcal{M}-1|}^2]$ will be mutated to select different technicians, while the elements $[c_0^1 : c_{\eta-1}^1]$ and $[c_{\eta-1}^2 : c_{|\mathcal{M}-1|}^2]$ 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 $[\mathbf{e^2}, \mathbf{b^2}, \mathbf{c^2}]$.

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.

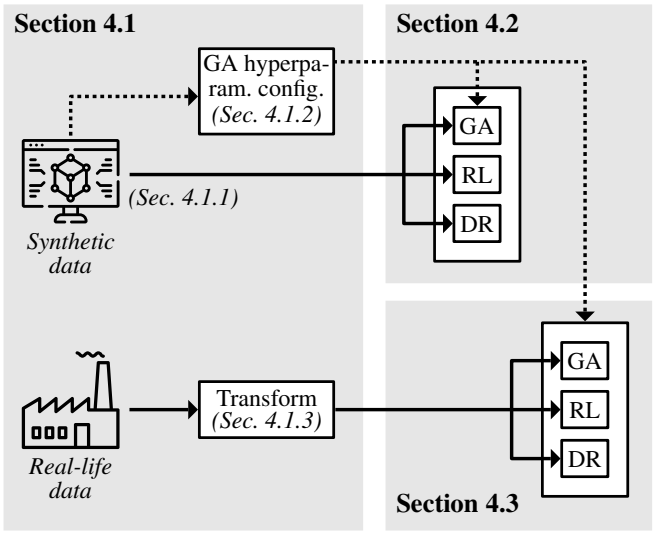


Figure 4.3: Schematic representation of the organizational structure in Section 4.4

4.4 Result and analysis

In this section, we present the evaluation of DRL, GA-S, and DR to solve the dynamic scheduling problem previously defined (cf., section 4.3.1), along with a financial discussion. To ease understanding of how this section is structured, we refer the reader to Figure 4.3: section 4.4.1 discusses the experimental settings, including how artificial/synthetic data is generated to consider different degrees of uncertainty (section 4.4.1), how GA's hyperparameters are chosen (section 4.4.1), 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 Figure 4.3, both datasets (synthetic and real) are used as inputs of our experiments, whose results are presented and discussed in section 4.4.2. Section 4.4.3 presents a case study on use of a manufacturing industry. Section 4.4.4 presents a brief discussion of the financial considerations to apply each policy in terms of the uptime of the machines.

4.4.1 Experimental settings

Synthetic data

In reliability, the Weibull distribution is a continuous probability distribution commonly used to model the time to failure [SS22]. It can be defined by 2-parameters, shape (β) and scale (α) parameters as shown in 4.10.

$$f(x;\alpha,\beta) = \frac{\beta}{\alpha}\left(\frac{x}{\alpha}\right)^{\beta-1} e^{-(x/\alpha)^{\beta}} \tag{4.10}$$

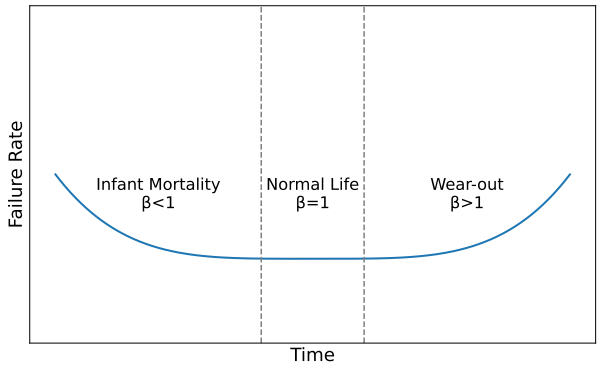
Table 4.2: Parameter Configuration for GA-S

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

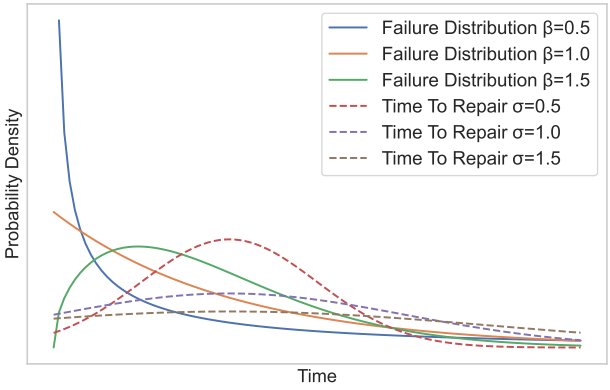
While the scale parameter represents a characteristic life, the shape parameter can characterize a specific type of failure. This type of failure can be observed in the bathub curve, where a $\beta < 1$ represents infant mortality, $\beta \approx 1$ a random failure, and $\beta > 1$ a wear-out effect [JM21], as shown in Figure 4.4a. In our experiments, two degrees of uncertainty (low and high) for the failure distribution and TTR 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 components where the uncertainty decreases as $\beta \to \infty$, reducing the spread of failures. The uncertainty in the TTR is defined as $\sigma \in \{0.5, 1.5\}$, where the uncertainty decreases as $\sigma \to 0$. These values were chosen to provide a streched distribution, indicating a 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 Figure 4.4b. Note that the proposed MDP has an horizon of 168 steps, equivalent to one week.

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 criterion (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 is the result of the combination of 3 Population size $\times$ 4 Mutation Rate $\times$ 3 Crossover Rate $\times$ 9-step configurations. In each of these configurations, the values of the objective function are normalized to facilitate 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 4.2 summarizes the different hyperparameters selected.

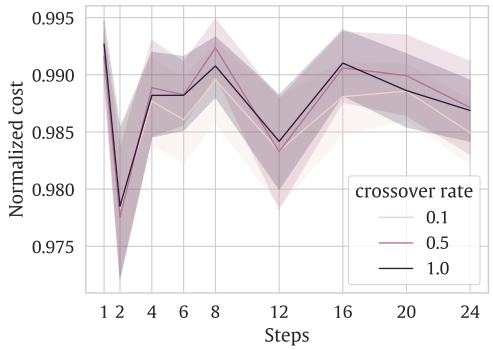


(a) Bathtub curve representing different failure rates over time

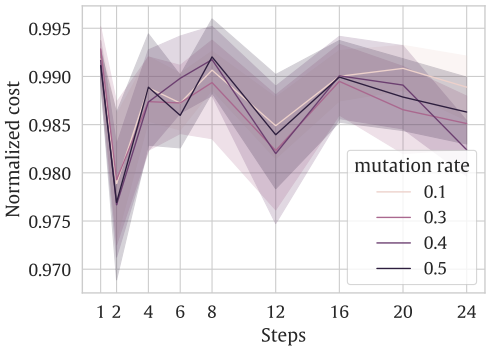


(b) Failure distribution and TTR with varying β, and σ parameters

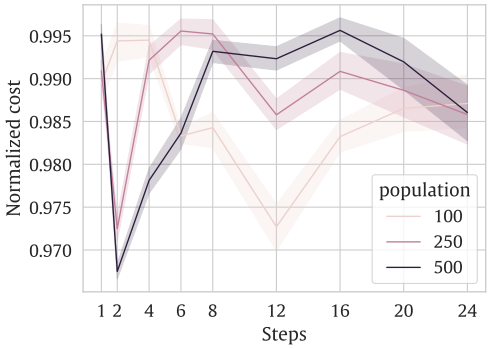
Figure 4.4: Illustration of failure rate and their corresponding phases in product life and model parameter uncertainty



(a) Performance of crossover rates



(b) Performance of mutation rates



(c) Performance of population sizes

Figure 4.5: Comparative analysis of genetic algorithm parameters: (a) Crossover Rates, (b) Mutation Rates, and (c) Population sizes, over 24 steps

In the GA-S approach, it is essential to define a specific re-optimization period to perform the rescheduling. In this sense, 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 to refer to a day (1 to 24), which means that one timestep refers to 1H in our environment. Figures 4.5a, 4.5b, 4.5c 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. Figure 4.5a 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

Table 4.3: 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

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 Figure 4.5b, 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, Figure 4.5c 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 higher complexity of the problem, a larger population size becomes more effective as it covers a wider range of solutions. In summary, Table 4.3 gives insight into the best configuration results per re-optimization period (G).

Real data transformation

In this section, we conduct the experiments in 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.

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

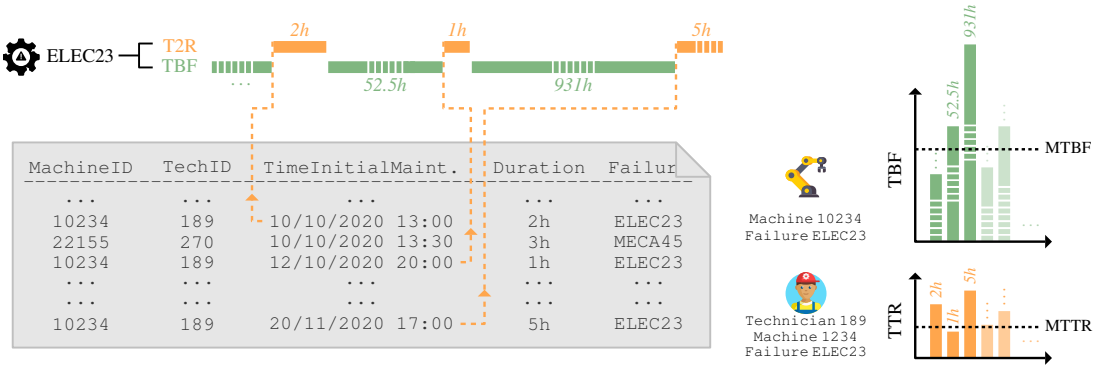


Figure 4.6: Illustration of feature extraction methodology based on maintenance data

μ_{ij}, σ_{ij}, α_{ij}, β_{ij} that will be used to represent the dynamic scheduling environment. Maintenance data is used to calculate the MTTR and Mean Time Between Failures (MTBF), as illustrated in Figure 4.6. These 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 subdivide $\mathcal{D}$ 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 is determined by $s_{k+1} - (s_k + c_k)$ where $k \in \mathcal{D}_{fj}$ (see green samples in Figure 4.6). To model the failure distribution, we fit a two-parameter Weibull distribution using the `Reliability` Library where we subsequently extracted the parameters α and β. For the 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 calculated based on the duration c_k, with these samples highlighted as orange in Figure 4.6. From this procedure, we calculated the average repair time μ_{fj} and the standard deviation σ_{fj} for each failure and technician combination.

4.4.2 Comparative analysis

In this section, the results of the comparative analysis of the methods (DRL, GA-S, DR) are presented and discussed. Let us remind ourselves that (i) the performance evaluation of each method is performed based on Equation 4.4 and considering the MDP defined in Section 4.3.1. The reader can find hereinafter a few more experimental setting information for each method:

- *DRL:* the DRL agent was trained using the PPO algorithm [SWD+17] 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

Table 4.4: Comparative performance evaluation of DRL, DR, GA, under different levels of uncertainty in Failure Distribution and Maintenance Duration

Uncertainty		**Methods**											
Fail Dist	Maint Dur	DRL		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

Table 4.5: Wall-clock time (s) comparison of DRL, GA, and DR for maintenance scheduling under differing degrees of uncertainty

Configuration		**Methods**					
Failure Distribution	Maintenance Duration	DRL	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

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 hours on 4 GPUs of Tesla V100-SXM2-32GB 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 4.3.1.

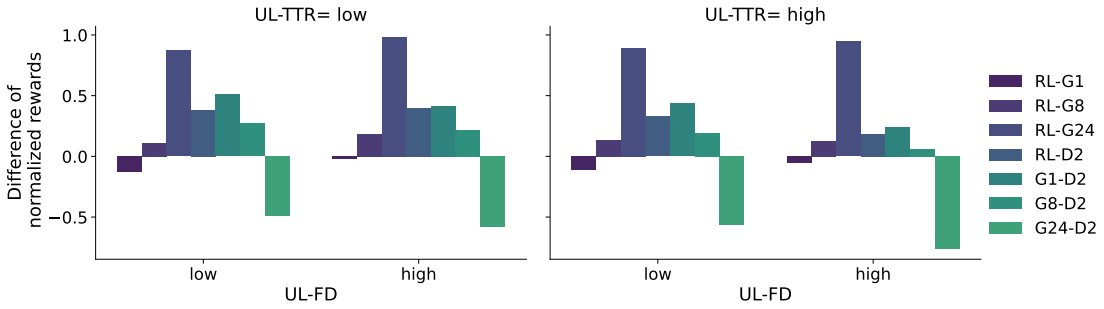


Figure 4.7: Performance differential of DRL, DR, GA across low and high uncertainty in TTR and Failure Distribution

The results of the comparison analysis are presented in Table 4.4 for the three

methods[2]. We used the reward and the MTTR as KPI 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 measures the working state of the machines and the remaining maintenance time. In the case of the reward (Eq. 4.4), the results obtained from DRL and G1 are quite similar; the total reward obtained is slightly higher in GA-S than in DRL for the cases where the uncertainty is low for the failure distribution, but the performance of DRL – *compared to GA-S* – increases along with the increase of uncertainty in the failure distribution. This finding can be visualized in Figure 4.7 where we display the difference between DRL, D2, and GA-S for 1 step (every hour), 8 steps (every shift), and 24 steps (every day). The greatest impact on the reward is influenced by the uncertainty of the Failure Distribution. With the DRL 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-S approach shows the opposite response under similar conditions. On the other hand, when the uncertainty in the Failure Distribution is high, it's the GA-S methods that improve the rewards, while DRL 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, DRL 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 DRL 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.

Let us now analyze the wallclock time it takes to make inferences for each of the methods, whose results are given in Table 4.5 (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 DRL, and finally GA-S. The wallclock time taken by GA-S to obtain the results for each step (1h step) is higher (≈ 1.72 on average) than the time needed to calculate the steps (1h), which makes it not applicable in real-time settings with our current computational

[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, 2h, 4h,. . . , 24h).

Table 4.6: Comparative Evaluation of reward, TTR, and solving time for DRL, DR, GA methods using real data

Metric	DRL	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	10476.30	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

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 hours (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 hand, 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.

4.4.3 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 4.4.1, 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 that they are satisfying specific criteria. Among other criteria, a shape parameter greater than 1.0 and the distributions with a low-scale parameter are expected, which eases the execution of the experiments, since failure distributions occur in a short amount of time modeling 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 4.6, reveal important insights into 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 hours reschedule) offers optimal benefits in terms of wallclock time, as frequent rescheduling does not provide significantly benefits.
- **DRL** once trained, shows a shorter inference time compared to GA-S. This efficiency translates into significant time savings when used recurrently, as illustrated in Figure 4.8. Furthermore, the results presented show that the reward is considerably higher compared to the other methods, indicating how well DRL methods deal with real data distributions to keep machines in a working state.
- **DR** present lower reward than compared to 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 DRL. 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 DRL 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 methos presents benefits and challenges that the decision maker should consider.

4.4.4 Financial consideration

The objective of the proposed model is to minimize the downtime of the equipment that leads to a reduction in 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 be maintained because it reaches its useful life, a spare part

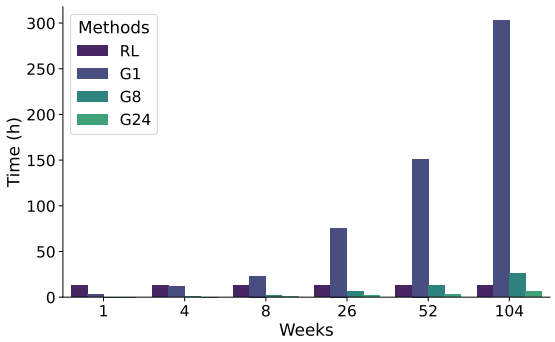


Figure 4.8: Wall-Clock time required by DRL, G1, G8, G24 for scheduling over different periods of time

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 MTTR but also maximizing 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.

4.5 Conclusion

Maintenance scheduling implies dealing with different types of uncertainty, spanning from the maintenance duration (which may highly vary depending on the assigned technician) to RUL estimation/approximination 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, DRL is increasingly used in all sectors due to democratization of AI/ML.

The present chapter covers 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 of increasing machine uptime and reducing MTTR, 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 DRL 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 TTR compared to DRL and GA-S. Although the exploration space is large, the complexity of the environment is low, allowing it to outperform in terms of MTTR.

In the following chapter, we will investigate how the design of DRL-based policies can overcome the classical maintenance policies used in the industry for environments with multicomponent machines. To do this, we will analyze the operational state of the equipment with the goal of maintaining high uptime while maximizing the RUL.

5

Multi-agent Deep Reinforcement Learning based Predictive Maintenance on Parallel Machines

Machines on the shop floor consist of multiple components that can fail unexpectedly, requiring technicians with varying levels of experience and skills to restore operations. In this chapter, we present a multi-agent reinforcement learning approach to address the optimization of maintenance and evaluate their performance against classical maintenance strategies.

Contents

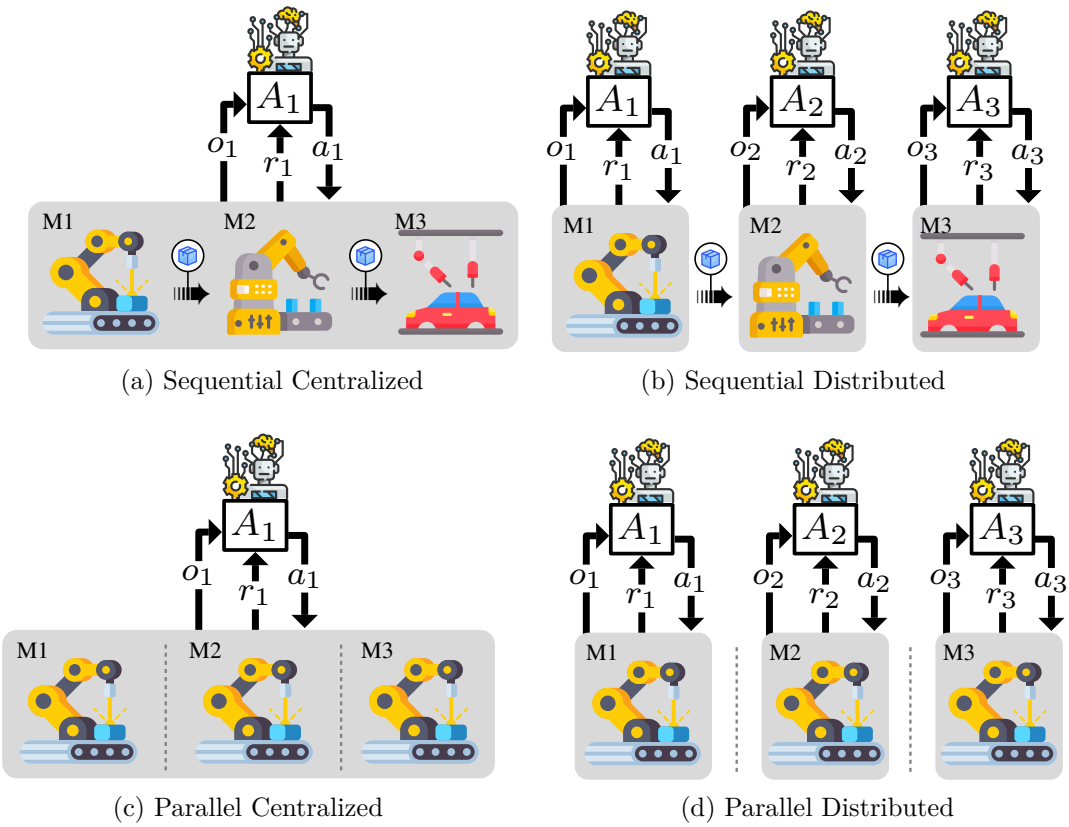
5.1 Introduction

In recent years, the use of ML techniques has gained significant traction in PdM applications in the manufacturing industry [SAS21]. While both supervised and unsupervised learning techniques have already been widely used in the manufacturing industry, accounting for 90-95% of all applications according to [DB21], especially for PdM [LPB+20], DRL has been much less studied [dGMO+21]. This limited focus on DRL is a missed opportunity, as DRL provides many interesting features [SB18], such as learning by interacting with the environment, measuring the utility of actions that yield long-term benefits, optimizing complex sequential decisions under uncertainty, adding that DRL takes advantage of the multi-agent approach, which allows for multi-objective optimization [KLS+20].

5.1.1 DRL-based Maintenance Policies and Strategies

RL-based maintenance strategies usually address a *sequential* or *parallel* manufacturing process. The former (sequential) refers to processes where multiple machines are connected and dependent on each other, as depicted in Figure 5.1a and 5.1b (machine M3 requiring, as inputs, goods manufactured by M2, which itself requires goods manufactured by M1), while the latter (*parallel*) refers to processes where machines are independent of one another (i.e., goods being manufactured by machines in parallel), as depicted in Figure 5.1c and 5.1d. For each type of manufacturing process, two types of agent architectures can be applied for decision-making, namely *centralized* (decisions regarding the different components/processes are taken by a single logical agent) or *distributed* (decisions are distributed among multiple logic agents), thus leading to the four architectural design possibilities depicted in Figure 5.1.

In the domain of centralized agent architectures, several approaches have been developed with the objective of optimizing maintenance scheduling, particularly in the context of parallel manufacturing environments where machines operate independently, as shown in Figure 5.1c. Xu and Cao [XC19] propose a centralized approach to schedule maintenance tasks of a machine tool, the objective being to improve the energy efficiency of the production process. This policy can be applied to different states of machine degradation, taking into account productivity, product quality, and energy consumption. A control policy for manufacturing facilities linked to a finished products buffer is proposed by Xanthopoulos et al. [XKK+17], where the agent policy allows to alternate between production and maintenance actions. This approach was compared with other production policies, such as `Kanbas` and `(s,S)`, along with maintenance policies such as CBM and periodic-based. Experiments have shown that the DRL-based policy succeeds in reducing the cost function, providing high-level of service while keeping inventory as low as possible. An extension of this work was proposed by Paraschos et al. [PKK20] to



(a) Sequential Centralized

(b) Sequential Distributed

(c) Parallel Centralized

(d) Parallel Distributed

Figure 5.1: Possible agent architectural designs when using Reinforcement Learning (RL) in industrial processes

include quality data that are related to the level of system degradation. In addition, the authors include a recycling policy so that agents can additionally perform the action of recycling second-class goods. This system is guided to maximize profits for the different products, and to minimize maintenance and production costs. Their work was tested in different scenarios with variable production times and deterioration frequencies. The results showed that the proposed policy achieves higher profits than other policies, and is effective in managing inventory levels and preventing equipment deterioration. Wang et al. [WYW21] propose optimization of production scheduling and maintenance of multiple factories in which a collaborative maintenance policy is performed. In their model, both machine deterioration and unexpected failures are taken into account with the aim of maximizing the total profit, where collaborative maintenance is ensured through the use of blockchain technology.

In the context of sequential manufacturing processes, where machines are

interconnected and dependent on each other, centralized approaches have proven effective in managing maintenance tasks, as illustrated in Figure 5.1a. For example, Huang et al. [HCC19] have been working on the problem of PM for a serial production line with multi-stage machines, this work having been extended in [HCA20] to find the optimal time to execute preventive actions. The authors have compared their work with respect to different maintenance policies such as CM, PM, opportunistic, and different DRL algorithms, whose results show that the DRL policy manages to keep maintenance and production costs below the other policies.

In the domain of distributed maintenance strategies, where multiple agents can collaborate to optimize maintenance tasks in different manufacturing environments, as presented in Figure 5.1b and Figure 5.1d. Kuhnle et al. [KJL19] focused on a parallel stochastic production environment to determine the best window of opportunity to perform maintenance. Their work is based on the use of multiple independent agents that learn the policy based on information from the production system buffer and time-to-failures. The DRL policy outperforms the CM and PM policies with respect to completed jobs and evidenced how the agents learn to execute maintenance closer to failure times with a low buffer volume. On the other hand, Su et al. [SHA+22] extended Huang et al. work [HCC19; HCA20], transitioning from single-agent to multi-agent systems for PM in sequential, multi-stage production lines. Their main contribution is to demonstrate convergence towards better policies and to solve scalability issues by the joint action space of the single agent approaches. Although the above-discussed studies are interesting in many respects, they still suffer from two main limitations: (l1) they do not fully account for the dynamic availability and skill levels of technicians, which can significantly impact the efficiency of maintenance tasks; (l2) they remain restricted to managing only a single type of machine failure, whereas in real-world environments, multiple failure types need to be addressed simultaneously.

Addressing these limitations requires careful consideration of the underlying agent architecture. Although there is a tendency to adopt centralized architectures due to the inherent complexities of distributed frameworks, distributed approaches introduce a number of challenges. These include agent heterogeneity, communication, the definition of collective goals (cooperation), scalability, non-stationarity, and so forth [DR19; NNN20]. However, distributed approaches offer a more realistic and potentially powerful solution when these challenges are effectively addressed and managed. For this reason, this chapter explores a new multi-agent DRL-based maintenance policy aimed at overcoming the previously discussed limitations (l1, l2).

To address the challenges of developing effective and adaptive maintenance policies, this chapter proposes a novel DRL-based maintenance strategy aimed

at dynamically responding to unexpected failures, managing uncertainty, and optimizing system uptime.

Section 5.2 provides the formalization of the DRL model that underpins the proposed maintenance policy. Section 5.3 presents the experimental setup and evaluation of our system in two manufacturing scenarios. Finally, a discussion and conclusion are given in Sections 5.4 and 5.5 respectively.

5.2 Multi-agent DRL-based maintenance policy

This Section presents the mathematical formalization underlying our DRL system. Section 5.2.1 introduces the environment considering a set of identical-parallel machines subject to multiple failures. Section 5.2.2 presents the DRL-Framework explaining in detail the interaction that the agents have with the environment. All notations used in this work are summarized in Table 5.1.

Table 5.1: Notation used in this chapter

Notation	Description
M	Set of machines
C_m	Set of components of $m \in M$
f_c	Failure time of $c \in C_m$
T	Set of Technicians
r_{tc}	Repairing time by technician $t \in T$ given a component $c \in C_m$
$W(x; \alpha, \beta)$	2-parameter Weibull distribution
z_m	State of $m \in M$
g_t	State of $t \in T$
w_c	State of $c \in C_m$
l_c	Lifespan of component $c \in C_m$
e_m	Remaining maintenance time of m
r_m	Reward obtained by agent in charge of m
$\mathbf{o_m}$	Local observation obtained by agent in charge of m
$\mathbf{a}_m$	Action taken by the agent in charge of m
η	Reward penalty due to breakdown prediction
p	Total profit

5.2.1 System description

Let us consider a factory where the production is performed by M identical machines. Each machine $m \in M$ is integrated by C_m components, which can fail at $f_c \mid c \in C_m$ timesteps. The machine m can receive maintenance by any available technician $t \in T$ to restore or repair any $c \in C_m$. Each t has a repairing time r for

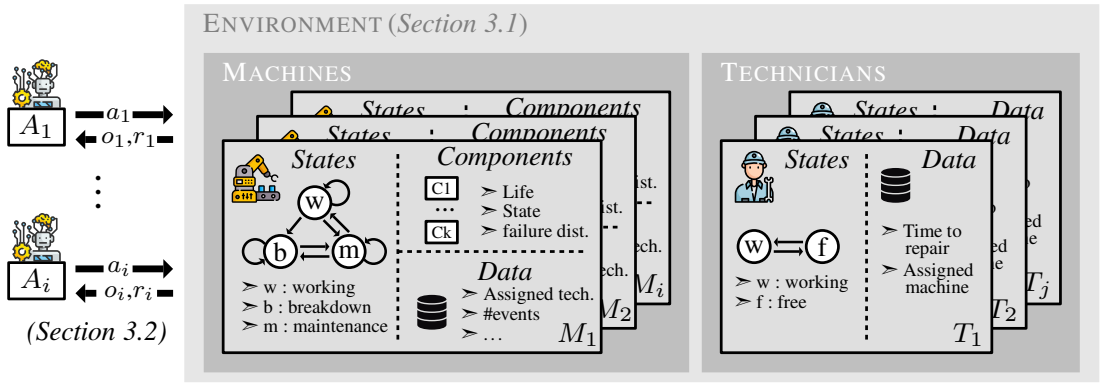


Figure 5.2: Environment proposed for multi-agent DRL-based PdM

each type of component c. The repairing time r by technician t given a component c is defined as $r_{tc} \in \mathbb{Z}^+$.

We can represent three states for each machine that evolve in discrete-time steps. The first is *working*, which indicates that the machine is running normally and has not shown any type of failure. The second is *breakdown*, in which one or more c have failed and the machine can not return to the *working* state until c has/have been repaired. To model the failure of a component, a 2-parameter Weibull distribution is used, as commonly adopted in the literature to describe equipment failures [Hal93], where the probability density function is defined as $W(x; \alpha, \beta) = \frac{\beta x^{\beta-1}}{\alpha^\beta} e^{-(\frac{x}{\alpha})^\beta}$, with α the scale parameter and β the shape. The last state is *maintenance*, which indicates that a technician t has been assigned to repair a c in r_{tc} time steps. If the machine does not present any other type of failure, then it will switch back to the *working* state. Ideally, the machine should move from the *working* state to the *maintenance* state without having to go through the *breakdown* state. Finally, any available technician can be assigned when the machine is in the *working* or *breakdown* state.

5.2.2 Multi-agent models

In order to obtain optimal or near-optimal maintenance policies, a DRL approach is employed. According to this, multiple decision-making agents are placed in an environment whose dynamics are initially unknown [SB18]. This environment can be formally described using an extension of the MDP called Markov Games (MG), also known as Stochastic Games. As mentioned in Chapter 4, an 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. MDP can be expressed as a 5-tuple $\langle S, A, \mathcal{P}, R, \gamma \rangle$, which consists of a state space S indicating all possible states the agent can be in; an action space A indicating

all possible actions the agent can take; a transition function $\mathcal{P} : S \times A \to S$ indicating the probability of transitioning from any state $s \in S$ to state $s' \in S$ given that the agent took action $a \in A$; a reward function $R : S \times A \times S \to \mathbb{R}$ that returns an immediate reward given by the transition from (s, a) to s'; and a discount factor $\gamma \in [0, 1]$ indicating how myopic the agent is, a $\gamma = 0$ indicating that the agent only cares about immediate reward and in the case of $\gamma \to 1$ the agent gives more weight to future state information. MG is a generalization of MDP to work with multiple agents. MG for N agents can be defined by the 6-tuple $\langle N, S, \{A_i\}_{i \in N}, \mathcal{P}, \{R_i\}_{i \in N}, \gamma \rangle$, which consists of a state space S indicating the state space observed by all the agents; an action space A_i indicating all possible actions the agent i can take; a transition function $\mathcal{P} : S \times A \to S$, given that $A = A_1 \times ... \times A_i$, indicating the probability of transitioning from any state $s \in S$ to any state $s' \in S$ for any joint action $a \in A$; a reward function $R_i : S \times A \times S \to \mathbb{R}$ that returns an immediate reward received by agent i for a transition from (s, a) to s'; and a discount factor $\gamma \in [0, 1]$.

The proposed predictive maintenance policies (DRL policies) performed two tasks: (i) keep the machines in a *working* state as long as possible; (ii) prevent any failure that could lead to a *breakdown* state by triggering a maintenance action. Using the MG framework, agents determine their actions based on the observations given by the system. Therefore, it is required to design local observations that provide the basis for the agent's actions and provide useful information for training. To perform the above described tasks, agents observe a representation of the system state that includes the state of all the machines and technicians. However, each agent is limited to observe the lifetime of the components of its own machine and any remaining time of maintenance. This limits the decision-making of the agents (since they are partially unaware of the complete state of the other machines), which contributes to decrease the complexity of their decision-making.

The system state observed by an agent a_m responsible for the machine m is represented by the vector given in (5.1), where $z_i \in \{0, 1, 2\}$ gives information whether machine $i \in M$ is in a *working*, *breakdown* or *maintenance* state respectively, $g_j \in \{0, 1\}$ whether a technician $j \in T$ is working on a particular machine (0) or free (1), w_c represents the state of component $c \in C_m$, l_c the lifespan of component $c \in C_m$, and e_m the remaining maintenance time of machine m.

$$\mathbf{o}_m = (z_1, ..., z_{|M|}, g_1, ..., g_{|T|}, w_1, ..., w_{|C_m|}, l_1, ..., l_{|C_m|}, e_m) \tag{5.1}$$

Depending on the observation, each agent must select whether to perform a maintenance action by selecting an available technician and the component to be maintained or to delay maintenance. The action of agent $\mathbf{a}_m$ responsible for machine m is given in (5.2) where d means that no technician will be assigned to m.

$$\mathbf{a}_m \in T \times C_m \cup \{\mathbf{d}\} \tag{5.2}$$

To guide agents in learning the policy, a reward function is defined to benefit the agent to keep the machine in the *working* state and to predict the best time to trigger maintenance actions. This function is defined in (5.3), where $\mathbf{o}_m$ is the observation of the current state of the machine m, $\mathbf{a}_m$ the action taken, and $\mathbf{o'}_m$ the next observation obtained from machine m. η is defined in (5.4), with $f_c \sim W(x; \alpha, \beta)$ the time steps in which the selected component c by the maintenance action will fail, and l_c the current time steps of the component.

$$r_m(\mathbf{o}_m, \mathbf{a}_m, \mathbf{o'}_m) = \begin{cases} 1 & \text{if } \mathbf{o'}_m = working \\ 1-\eta & \text{if } \mathbf{o'}_m = maintenance \\ 0 & \text{if } \mathbf{o'}_m = breakdown \\ -2 & \text{if } \mathbf{a} = invalid \end{cases} \tag{5.3}$$

$$\eta = \begin{cases} \frac{f_c - l_c}{f_c + l_c} & \text{if } f_c > l_c \\ 1 & otherwise \end{cases} \tag{5.4}$$

Overall, the agents learn how to keep machines in the *working* state as long as possible due to the fact that this condition allows them to obtain the highest reward and, indirectly, to identify the best technicians to perform maintenance in the shortest amount of time steps to move out of the *breakdown* and *maintenance* state as quickly as possible.

5.3 Evaluation

For evaluation purposes, the proposed DRL-based maintenance policy will be compared to traditional ones, such as CM and PM. Section 5.3.1 describes such policies. Section 5.3.2 presents the experimental setup. Section 5.3.3 presents the training phase of the multi-agent system. Section 5.3.4 presents and discusses the results obtained.

5.3.1 Maintenance Policy Benchmarking

To evaluate the proposed DRL-based maintenance policy, three maintenance policies are implemented: corrective (CM), preventive (PM), and random (RA), in a way similar to that proposed in [KJL19]. Each of these policies is described below.

CM is the simplest policy. It is based on waiting for any of the components c in machine m to fail before performing the maintenance action. Once the machine goes to the *breakdown* state, it checks if there is a technician t available. If any, technician t is assigned to m, otherwise, m makes the t query at each time step

until it can perform the assignment. The pseudocode of this policy, which has been implemented in our environment, is detailed in Algorithm 1.

Algorithm 1 CM policy

```
m ← current machine
C_m ← set of Components
if state(m) = breakdown then
    for t in T do
        if state(t) = available then
            for c in C_m do
                if state(c) = break then
                    do maintenance to c by t
                    break
```

The RA policy is part of the preventive maintenance policy because it tries to perform maintenance actions before the failure occurs. For this, an ϵ can be defined, which determining the probability of requesting a technician t to perform maintenance actions on any of the components $c \in C_m$, even if maintenance is not 'necessary' (i.e., not critical). In case there is no technician available, the operation is repeated for the next time step. The pseudocode of this policy is detailed in Algorithm 2.

Algorithm 2 RA policy

```
Parameters ε ∈ (0, 1]
m ← current machine
C_m ← set of Components
r ← random number between 0 and 1
c_r ← random c from C
if r < ε then
    for t in T do
        if state(t) = available then
            do maintenance to c_r by t
            break
```

The PM policy performs maintenance actions periodically. A time period $\mathcal{T}$ should be set for each $c \in C$. This will indicate the times at which maintenance will be requested on the components $c \in C$. In case that c failed before the maintenance action is performed, the PM policy can request a technician t to perform a corrective action to the particular component that failed. The pseudocode of this policy is detailed in Algorithm 3.

Algorithm 3 PM policy

Parameters $\mathcal{T} = \{\tau_1, ..., \tau_c\}$ period of maintenance for each c in C_m
$i \leftarrow$ current timestep
$m \leftarrow$ current machine
$C_m \leftarrow$ set of Components
if state(m) = working **then**
  for τ in $\mathcal{T}_c$ **do**
    if i = τ **then**
      for t in T **do**
        if state(t) = available **then**
          do maintenance to c by t
          break
else
  do apply the CM Policy
  break

Table 5.2: Parameters of the studied scenarios

	$\|M\|$	$\|T\|$	$\|C_m\|$	α_c	β_c	r_{tC}
Scenario 1	3	2	2	$\langle 6, 8\rangle$	$\langle 5, 5\rangle$	$\langle (8, 2), (2, 8)\rangle$
Scenario 2	5	3	2	$\langle 6, 8\rangle$	$\langle 5, 5\rangle$	$\langle (8, 2), (2, 8), (8, 8)\rangle$

The next Sections detail the experimental environment and setup in which the three above maintenance policies and ours (RL) have been implemented for evaluation and comparison purposes.

5.3.2 Experimental setup

To test our system, a variable number of agents and technicians is defined and two types of component failures are modeled based on the $W(x; \alpha, \beta)$ distribution. A different α for each type of failure is used, which analogously represents the time to failure, and a single $\beta \geq 2$ is set to model wear-out failures [DS98]. Three technicians are defined, where only the first two are used in the first scenario (as reported in Table 5.2). The repair time of the technicians is set so that one technician is highly experienced in repairing one type of failure (short repairing time) and inexperienced in repairing the other (long repairing time), the opposite applies for the second technician. The third technician is inexperienced in repairing both types of failures. To evaluate the proposed policy with respect to the policies described in Section 5.3.1 (CM, PM, RA), the horizon of the episode is limited to 1.5 times the maximum 90^{th} percentile of the failure distributions. In this way, we

can evaluate the occurrence of both failure distributions in a short period of time. The configuration used for each scenario (3 machines and 5 machines) is given in Table 5.2.

5.3.3 Training of DRL agents

To evaluate our system, agents are trained using the Proximal Policy Optimization algorithm [SWD+17] based on the `RLlib` library implementation. This 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. Additionally, we use Curiosity [PAE+17] as an exploration strategy to address the lack of diversity in reward (sparsity) due to the fact that some actions may not provide an immediate reward (e.g., delay action). Curiosity is a type of intrinsic motivation for exploration that encourages the agent to experience novel states. Deepak Pathak et al. [PAE+17] define curiosity as the error in an agent's ability to predict the consequence of its own actions in a visual feature space learned by a self-supervised inverse dynamics model. Agents are trained with completely independent policies for 5 million steps. A fully connected 2-layer of 16×16 is used, a batch size of 2000, and a learning rate of 1e$-$3, while keeping the remaining hyperparameters by default. Figure 5.3a and 5.4a show the training curves (green curves) of the 3M and 5M scenarios, respectively, and their average (purple) over the $50k$ steps, where agents manage to learn the policy despite the non-stationarity effect of the learning of multiple agents.

5.3.4 Maintenance policy comparison analysis

As detailed in section 5.3.1, policies learned by our multi-agent system are evaluated with respect to the CM, PM, and RA policies. All the actions of these policies are valid, so unlike DRL agents, these agents know what each agent chooses when forming the joint action to be executed in the environment. This prevents, for example, multiple agents from choosing the same available technician, which DRL agents do not know *a priori* when performing the joint action. In the RA policy, as shown in Algorithm 2, an $\epsilon = .35$ is set, which prevents the agent from executing a maintenance action too quickly. For the CM policy, there is no parameter to be adjusted/set in Algorithm 1 since the agents wait until the machines switch to the *breakdown* state to request a maintenance action. In the PM policy, as shown in Algorithm 3, the maintenance action is triggered based on the failure distributions. In this respect, the maintenance request times are given by $\mathcal{T} = \{\mathbb{E}[f_1], ..., \mathbb{E}[f_{|C_m|}]\}$, where $\mathbb{E}[f_c]$ represents the expected value of the distribution of failures $f_c \sim W(x; \alpha, \beta)$ for component $c \in C_m$. In case there is a failure before requesting maintenance, then the CM policy is performed. To evaluate the different maintenance policies, four metrics are defined: percentage of time that the machines were in working state; performance based on the total

reward obtained for each policy; total number of breakdowns per episode; total number of maintenance actions per episode. In our experiments, the results were obtained by averaging the results over 1000 episodes.

The results obtained from our experiments are given in Figure 5.3 and 5.4 for the 3M and 5M scenarios. Looking at metric *percentage of time in working state* (see Figure 5.3b and 5.4b), results show that the RL policy outperforms the other three policies (CM, RA, PM), as RL allows machines to stay $\approx$ 60% of the time in the *working* state, against 40% to 50% for CM, RA and PM. In addition, RL spends $\approx$ 20% less time in the breakdown state than CM, RA, and PM, while spending a similar amount of time in the maintenance one. This brings evidence that the DRL agents not only succeed in predicting when to execute the maintenance actions to avoid a failure, but also in improving agent coordination (e.g., avoiding them to pick the same technician at the same time). Looking at metric *total reward*, as can be seen in Figure 5.3c and 5.4c for the 3M and 5M scenarios, the cumulative reward in the DRL policy outperforms the other policies between 20% to 35%. This suggests that because agents are guided by the reward to keep the machine in a working state, and they request maintenance actions as close as possible to the failures, it is a policy that reduces unexpected failures while reducing the collision of maintenance requests. The latter is due to the fact that, if several agents request at the same time the same technician, this is considered as an invalid action, which is penalized and causes a restart of the environment, thus preventing the maximization of rewards. Looking at *total number of breakdowns* (see Figure 5.3d and 5.4d), the policy learned by DRL agents – *being guided to obtain a reward for approaching the time step in which there will be a breakdown* – succeeds in reducing the number of breakdowns. An explanation lies in the fact that the other policies do not have adequate knowledge of when to trigger maintenance due to the lack of coordination between agents. This could be different for the case of the PM policy if $|T| \geq |M|$, since, on average, it would be executing the maintenance action taking advantage of the complete useful life of the components. However, this scenario is unrealistic considering a manufacturing scenario. Instead, if the value of ϵ in the RA policy is increased, this would decrease the number of failures but at the cost of an increase in the number of maintenance actions.

Figure 5.3e and 5.4e give insight into the *number of maintenance actions* performed by each policy. Interestingly, the RL policy generates a higher number of actions. However, this needs to be put into perspective considering how this contributes to reducing the overall system downtime, for which RL outperforms the three other policies (as previously analyzed with Figure 5.3b and 5.4b). One reason that might explain this result is that scheduling a higher number of maintenance actions can prevent machines from switching to the failure state, but also avoid any queue for the availability of technicians to trigger the maintenance action.

5.3.5 Financial implication

The different policies in both scenarios were evaluated from an economic point of view considering three key variables obtained from Section 5.3.4. Since the profit of the company is obtained by their productivity, we evaluate the total profit with respect to the number of time steps the machines are in the *working* state. However, given that maintenance actions and breakdowns also represent a cost to the total profit, these two parameters (number of breakdowns and maintenance actions) are also taken into account when calculating the total profit.

Based on this premise, the total profit is calculated using (5.5), where p represents the total profit, p_{max} the maximum profit, a the total percentage of time steps in the *working* state, b the number of breakdowns, c the number of maintenance actions, and $\alpha \in [0, 1]$ and $\beta \in [0, 1]$ a ratio of the maximum profit that each breakdown and each maintenance action costs, respectively. The maximum possible profit, per episode, is the scenario where the machines are all the time in the *working* state.

$$p = p_{max}(a - (\alpha.b + \beta.c)) \tag{5.5}$$

Figure 5.5a and 5.5b illustrate that the top-1 policies with different values of α and β represent between 1% and 10% of p_{max} for breakdown events and maintenance actions. These are extreme values, but they allow us to identify the frontiers where one policy outperforms another from a total-profit perspective. Dark colors (also denoted by "+" in the legend) represent a profit gain, while light colors (denoted by "−") represent a loss. For different values of α and β, the RL, RA, and PM policies perform better. However, unlike DRL and PM, the RA policy never leads to a positive profit, which is unrealistic. As can be seen from the results, if the cost per number of maintenance actions increases, the preventive policy becomes a better choice than the RL policy. On the other hand, if β is kept low, the RL policy is not affected by an increase of α, which is due to the prevention of breakdowns.

5.4 Discussion

Although the present research work opens up a new direction to maintenance task scheduling in uncertain environments and with the dynamic assignment of technicians with different skills, some limitations or improvement opportunities should be raised.

First, one of the great advantages of multiagent systems is the distribution of a complex task into multiple simpler tasks among different agents to avoid exponential growth of the joint action space [ZTZ+21; FFA+18]. However, multi-agent RL systems present different challenges such as agent heterogeneity, communication, the definition of collective goals (cooperation), scalability, design of compact representations of the true state of the environment, and the main problem of non-stationarity

[DR19; NNN20]. Many approaches propose to adopt a centralized-learning approach (*cf.*, Figure 5.1) to address some of these challenges [FFA+18; RSD+18]. Nevertheless, decentralized-learning has proven to be as effective in addressing this problem [dWGM+20]. In the current work, we use a fully decentralized learning paradigm of RL that may prove to be computationally expensive due to the large search space on a day-to-day basis, the dynamic nature of the manufacturing floor, and the interactions between agents that can become complex to be managed [MGC+22]. To overcome this, new approaches should be investigated. Combining RL with metaheuristics for the exploration of RL agents is a possible direction [Dru19]. Besides, we consider that it is also important to evaluate the design of the agents' observations with respect to the type of learning to be used.

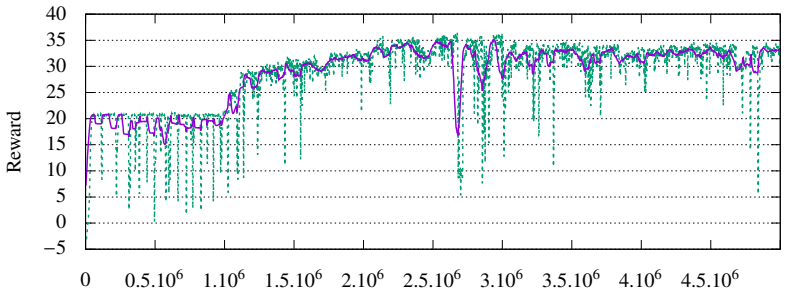
Second, for the assignment of technicians to perform a maintenance task, a planner is usually in charge of obtaining all the requirements, such as the number of technicians needed, the craftsmanship skills, the working hours, the duration of the job, *etc.* [Pal19]. However, when defining the policies obtained by the RL agents, these policies can exhibit (very) complex behaviors, which can cause distrust when planners, schedulers, or even technicians are part of the decision-making process [SPC+21]. Therefore, it is very important in future PdM frameworks to focus "explaining" the decisions resulting from such frameworks, which is also known as XAI (eXplainable AI) in the literature [MGC+22; MMS+20; JKF+19].

Finally, in the approach presented in this chapter, the maintenance policy learned by the DRL agents is based on equipment degradation, and on the skills of the technicians and their availability. However, as evidenced in section 5.1.1, in order to optimize the learned policy, it is important to explore the development of a joint policy with production activities to look for the best time windows of opportunity [KJL19] to perform maintenance actions, but also to take into account quality data (that is, monitoring how the quality of a given process degrades over time) [PKK20; HFR21].

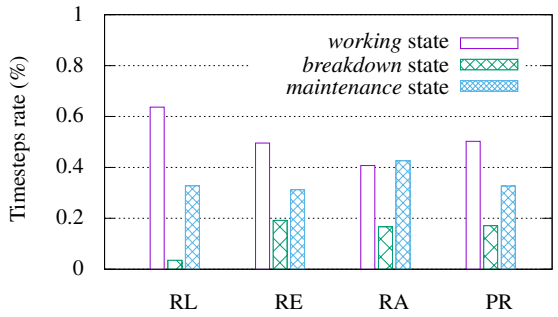
5.5 Conclusion

In this chapter we have examined a novel stochastic system that experiences multiple types of failures on a set of machines working in parallel (i.e., that are independent from one another). The problem consists in determining the best time to assign a maintenance task to a given technician (depending on her/his availability and skills) on a particular machine in order to avoid system failures and maintain a high operational uptime. To solve this problem, a multi-agent deep RL system is proposed, where each agent is responsible to monitor a single machine and trigger the (appropriate/optimal) maintenance action(s). The efficiency of the system is experimentally evaluated and compared with three other policies (reactive, preventive and random). Results of the proposed scenario show that the

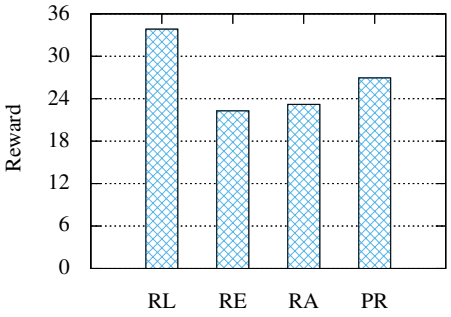
RL policy outperforms the others, reducing up to 80% the breakdown time, and up to 75% the failure prevention. To achieve these results, the RL policy performs more maintenance actions. Although in this chapter we have made an initial financial assessment to understand under which parameters one policy is superior to another, in the following chapter we will look more deeply at the economic as well as the environmental and social aspects of developing sustainable maintenance policies.



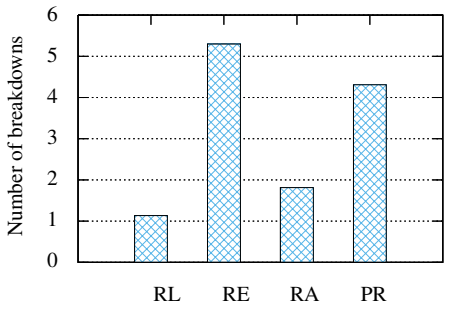
(a) Learning curve 3M



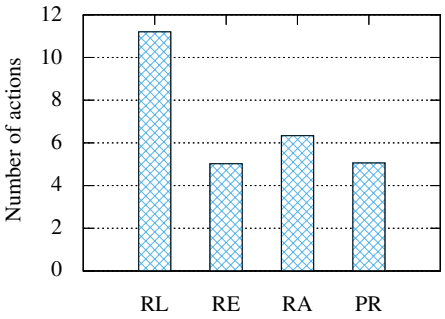
(b) Percentage of the timesteps in each state per policy



(c) Cumulative Rewards

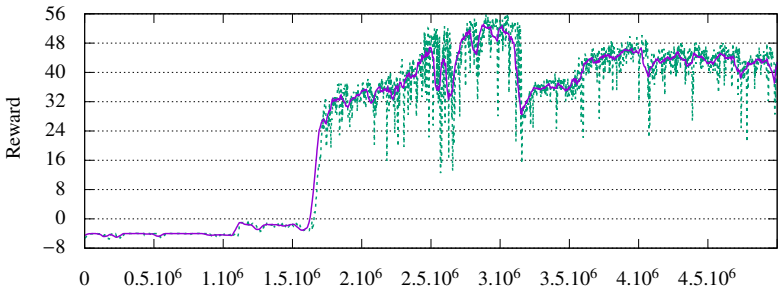


(d) Number of breakdowns per policy

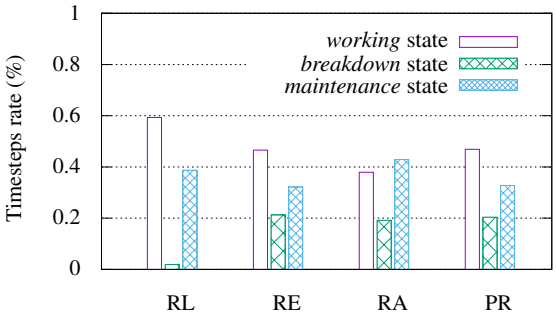


(e) Maintenance actions per policy

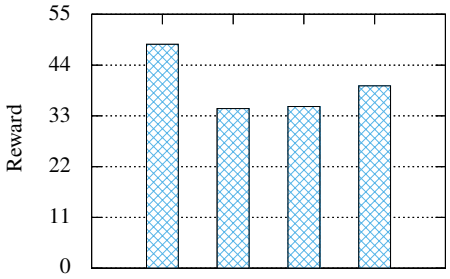
Figure 5.3: Training and evaluation of the 3M-Scenario



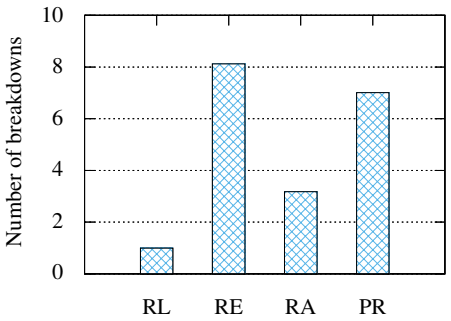
(a) Learning curve 5M



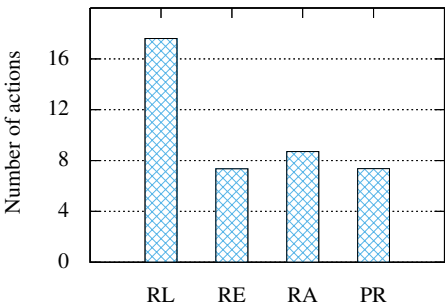
(b) Percentage of the timesteps in each state per policy



(c) Cumulative Rewards

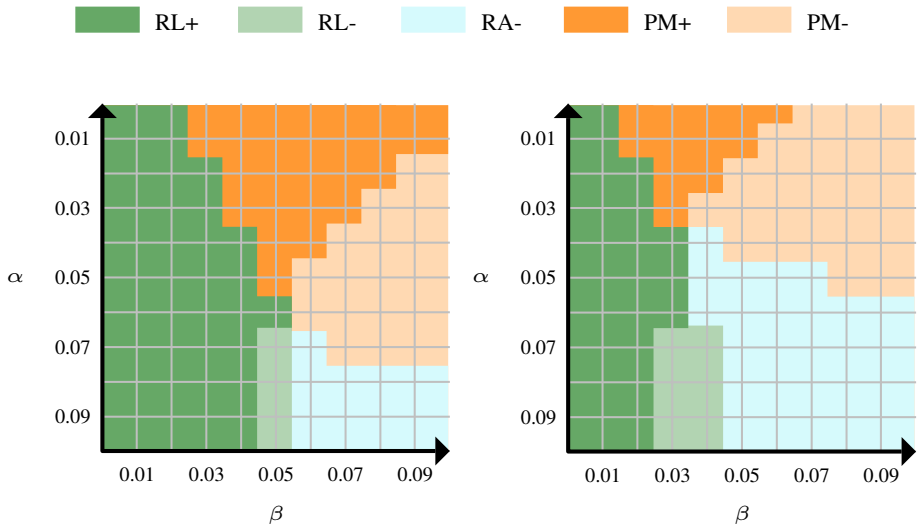


(d) Number of breakdowns per policy



(e) Maintenance actions per policy

Figure 5.4: Training and evaluation of the 5M-Scenario



(a) 3M-scenario (b) 5M-scenario

Figure 5.5: Top-profit policies for different α and β, which respectively represent the cost (percentage of the maximum profit) of a breakdown and maintenance action. Dark colors (denoted by "+" in the legend) represent a profit gain, while light colors (denoted by "−" in the legend) represent a loss.

6

Evolutionary multi-objective multi-agent deep reinforcement learning for sustainable maintenance scheduling

Sustainability has become a key objective for industry, including the manufacturing sector, where production and maintenance processes must now balance economic profitability with environmental and social responsibilities. In this chapter, we present the design of sustainable maintenance policies based on a new evolutionary reinforcement learning approach, which addresses the different objectives simultaneously.

Contents

6.1 Introduction

The development towards Industry 4.0 has a substantial influence on the manufacturing industry. It not only allows for establishing smart products and services, but also new and disruptive business models. Industry 4.0 technologies include, but are not limited to, standardized communication protocols, the Internet of Things, artificial intelligence, big data and analytics, blockchain, cloud computing, and simulation [WXZ+22]. Although these technologies are often used, in the first place, to improve production operations (aka Overall Equipment Effectiveness), their implication in the Sustainable Development Goals [Uni15] requires more attention and evaluation. In fact, traditional production systems are notorious for their poor ecological (and social) imbalances. It becomes a necessity for companies to address this problem for their future, as they face growing pressure from governments and customers to deliver sustainable products, aligned with European Green Deal-like initiatives [Eur19].

Predictive maintenance (PdM) moves away from traditional preventive maintenance (PM), where tasks are scheduled at regular intervals, or corrective maintenance (CM), where tasks are scheduled when a failure occurs, to optimal maintenance timing. This approach aims to maximize the overall profit of the manufacturing system [RZL+19]. This move comes with a dynamic maintenance task scheduling optimization problem [RKR+24], consisting of assigning a set of maintenance tasks to a set of technicians, while minimizing the overall system downtime and cost, and taking into account several constraints related to production, safety, technician skills, job duration, etc. [Lee96]. As illustrated in Figure 6.1, traditional scheduling optimization strategies such as CM, PM, or PdM focus primarily on economic indicators and rarely, if ever, consider the two other pillars of the Triple Bottom Line (TBL): Environment and Social. However, there is room to jointly optimize the three dimensions. Inspection and maintenance of energy-intensive machines during a period of time where the grid carbon intensity is high can contribute to reducing carbon emissions. Similarly, an employee-focused maintenance policy can deliver economic benefits by reducing productivity losses due to accidents or fatigue, while also having a positive impact on social factors such as employee morale. This chapter advances the state of the art in two significant ways:

- It is the first to consider both grid carbon intensity and technician fatigue in the maintenance scheduling optimization process, in addition to the economic criterion;
- It introduces an evolutionary multi-objective multi-agent deep Q-Learning (EvoDQN) maintenance scheduling approach, enabling companies to adjust, whenever needed, the importance and preference of each TBL criterion without incurring any re-optimization cost/time.

Section 6.2 discusses the pressing need for organizations to adopt sustainable

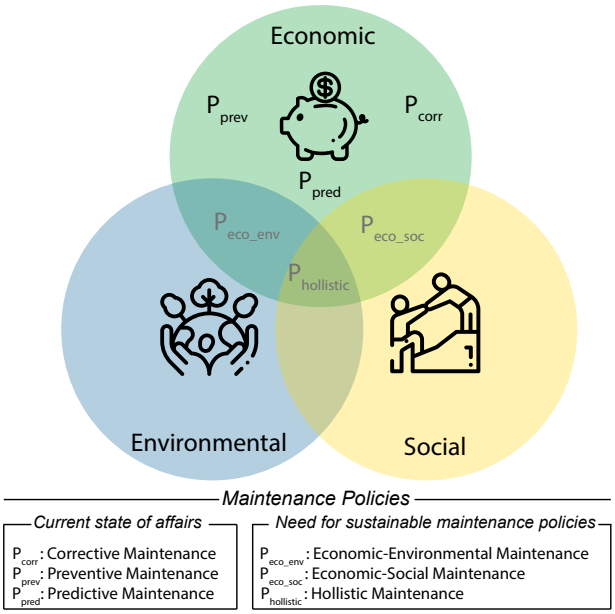


Figure 6.1: Maintenance policies from a TBL perspective

practices, providing an overview of the evolution of regulations and certifications over the years, and presents a realistic scenario in the manufacturing industry that requires the company to adapt its process due to customer requirements. Section 6.3 introduces EvoDQN. Section 6.4 evaluates and compares EvoDQN against traditional maintenance scheduling policies (corrective, preventive) and classical DQN approaches. Discussion and conclusion follow. All the variables used throughout the chapter are summarized in Table 6.1.

6.2 Sustainable manufacturing: a pressing need

Section 6.2.1 discusses the evolution of regulations and certification schemes over the years. Section 6.2.2 presents a scenario that illustrates the need for manufacturing processes to be flexible enough to accommodate real-time adjustments to the TBL criteria, resulting in diverse production and maintenance schedules.

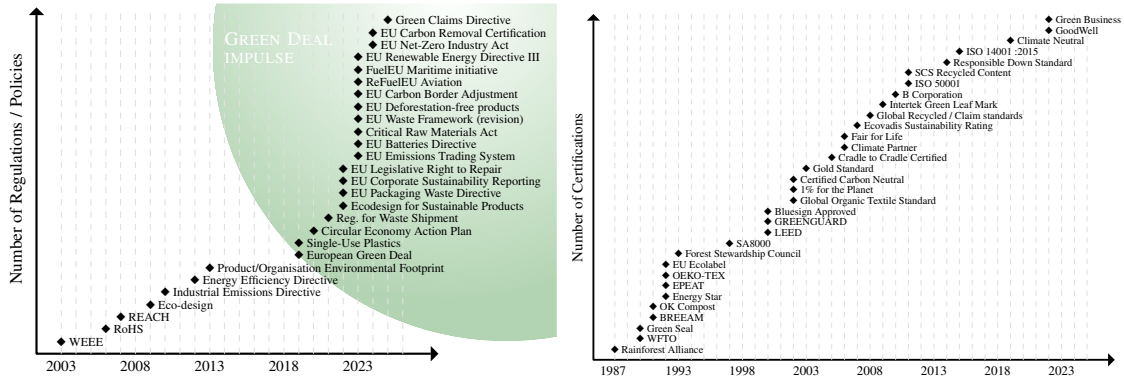
6.2.1 Evolution of regulations & certifications for sustainable product development

The manufacturing sector is under increasing pressure to adopt more sustainable practices, focusing on all aspects of the TBL [PMA23]. Recent literature and global initiatives show evidence of a willingness among companies and governments to improve the situation. The evolution of key certifications and regulations for sustainable development, depicted in Figure 6.2a and Figure 6.2b respectively,

Table 6.1: List of variables and their descriptions

Var.	Description
M	A set of machines: $M = \{1, .., m\}$
T	A set of technicians
E_m	Energy consumption (kWh) of machine $m \in M$
ω_m	State of machine for $m \in M$
ω_t	State of technicians for $t \in T$
$\Gamma_m(\alpha_m$, $\beta_m)$	Gamma process for machine $m \in M$, with shape α_m and scale β_m parameters
ρ_m	Failure threshold for $m \in M$
g_m	Degradation on machine m
q_t	Remaining time of technician t performing maintenance
μ_{car}	Mean of the normal distribution for the CO2 forecast parameter
σ_{car}	Standard deviation of the normal distribution for the CO2 forecast parameter
C^P	Perfect maintenance cost
C^I	Imperfect maintenance cost
C^D	Downtime cost
C^B	Breakdown cost
τ^B	Increased breakdown repair time
ξ	Weights for balance workload
$\mathcal{T}_{t,m}$	Maintenance time of technician t in mach. m
λ^F	Fatigue parameter
μ^R	Recovery parameter
μ_{break}	mean for the additional repair time after a breakdown
σ^2_{break}	variance for the additional repair time after breakdown
η	Carbon footprint forecast
η^n	Carbon footprint noise
η^s	Carbon footprint steps forecast
υ_t	Fatigue level of technician t
m_{HDL}	Machine with the Highest degradation level
m_{HCT}	Machine with the Highest cycle time
t_{mF}	Technician with the minimal level of fatigue
t_{MS}	Technician with the maximum skills
κ_p	Perfect maintenance
κ_i	Imperfect maintenance
S	State space
A	Action space
$\mathcal{P}$	Transition function
R	Reward function
γ	Discount factor

highlights a significant increase in regulations and certification schemes over the past two to three decades. Notably, Figure 6.2b shows an exponential increase in regulation starting from 2019, largely due to the European strategy – *through the "European Green Deal"* – to make the EU's economy sustainable. Under the



(a) Evolution of certifications

(b) Evolution of regulations/policies

Figure 6.2: Overview of the evolution over time of key regulations and certification for sustainable development.

emerging pressure for the industry to market products as "sustainable" or "green", it is expected that companies will increasingly subscribe to green certification schemes (issued by independent third-party organizations) to ensure their customers of the products' credibility. For example, adopting cornerstone certification schemes like ISO 14001 (Environmental Management System) is anticipated to provide a competitive advantage and increase firm value, as discussed in [WSU22]. While most regulations and certifications focus on environmental sustainability, a few also emphasize the social pillar, such as SA8000 (Social Accountability International), which certifies an organization's capability to meet standards for worker safety and well-being.

6.2.2 Motivation scenario for adjusting production and maintenance schedules to meet customers' sustainable development requirements

With the increasing pressure on companies to adopt sustainable development practices, they are constantly looking for ways to improve their products and processes. The 6R methodology [JDR+06] offers six key directions for improvement: (1) Reduce, (2) Reuse, (3) Remanufacture, (4) Recycle, (5) Reclaim or Recover, and (6) Redesign. Focusing on the first direction (Reduce), several strategies can be employed, such as using fewer materials and resources, reducing the number of components, minimizing pollution, or decreasing consumer returns. One approach to reducing pollution is to optimize production and maintenance schedules taking into account both environmental and social criteria. Figure 6.3 illustrates, in a comic strip format, a scenario where the manufacturing process is made flexible

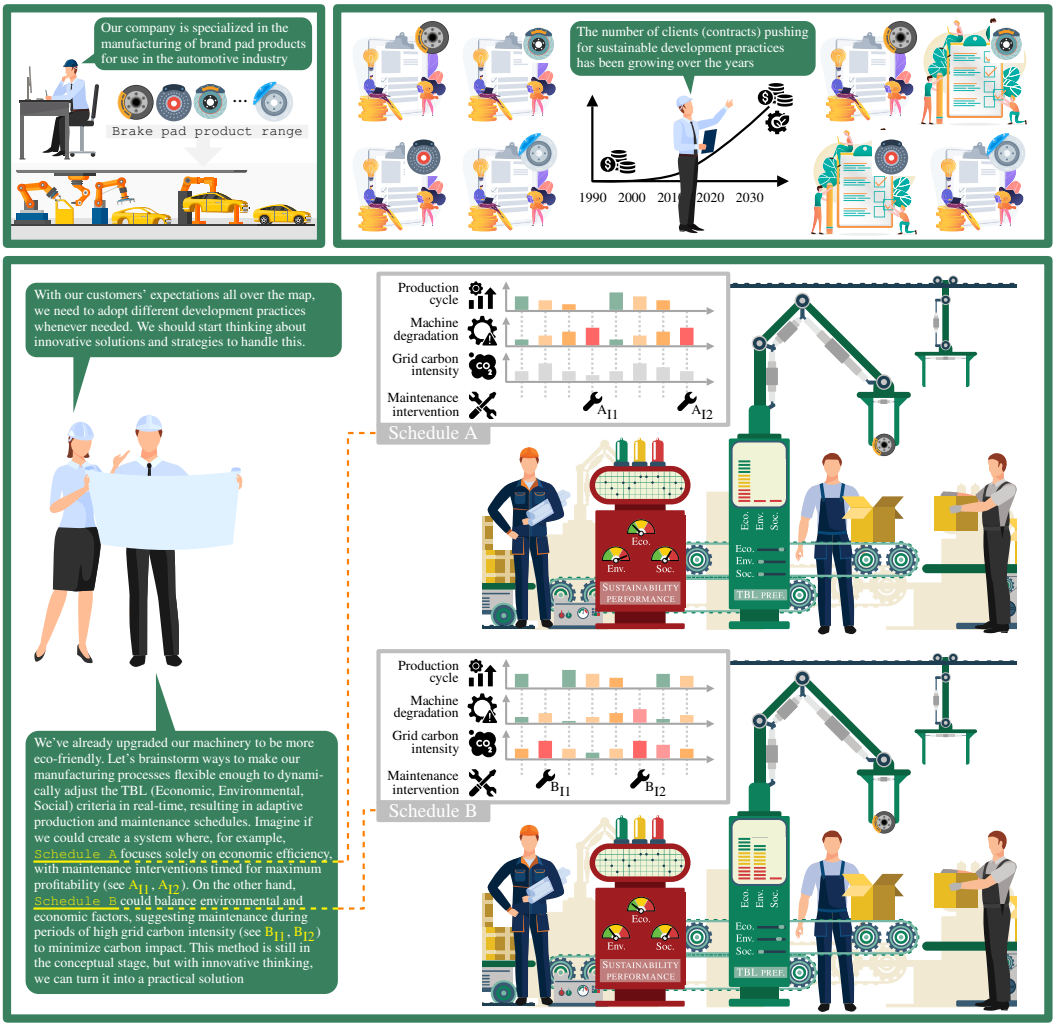


Figure 6.3: Illustration of dynamic adjustment of TBL criteria in manufacturing processes

enough to dynamically adjust the TBL criteria (Economic, Environmental, Social) in real-time, effectively meeting each customer's TBL needs. This illustration focuses on adjusting only the Economic and Environmental criteria to reduce the complexity of the scenario, but of course the social dimension (e.g., operator fatigue) is important to be considered too. This innovative strategy is the central proposal of this chapter.

6.3 EvoDQN for sustainable maintenance policies generation

This section presents the evolutionary multi-objective multi-agent DRL (EvoDQN) approach for sustainable maintenance scheduling, which allows a maintenance manager or decision maker to select – *when starting a new production batch* – a sustainable maintenance policy based on the specified TBL criteria preferences. These criteria are used to identify the closest policy among all those trained with EvoDQN, as detailed in Section 6.3.2. To facilitate understanding of the approach, Figure 6.4 illustrates the overall process. Initially, based on a predefined set of preferences (see frame a), multiple agents are created (see frame b). The overall goal is to enable agents to learn the policies that are distributed across the preference space. The agents' preferences are used to scalarize the reward during training. New agents will be created during the evolutionary process (see Evolutionary Operator in frame b) with the objective of maximizing hypervolume. The policies obtained with EvoDQN are then evaluated and normalized (see frame c), which allows the identification of the policy that aligns the most closely with the preference specified by the manager (see frame d) by the nearest neighbor (frame e). The best / selected policy (see frame f) is then deployed to the environment.

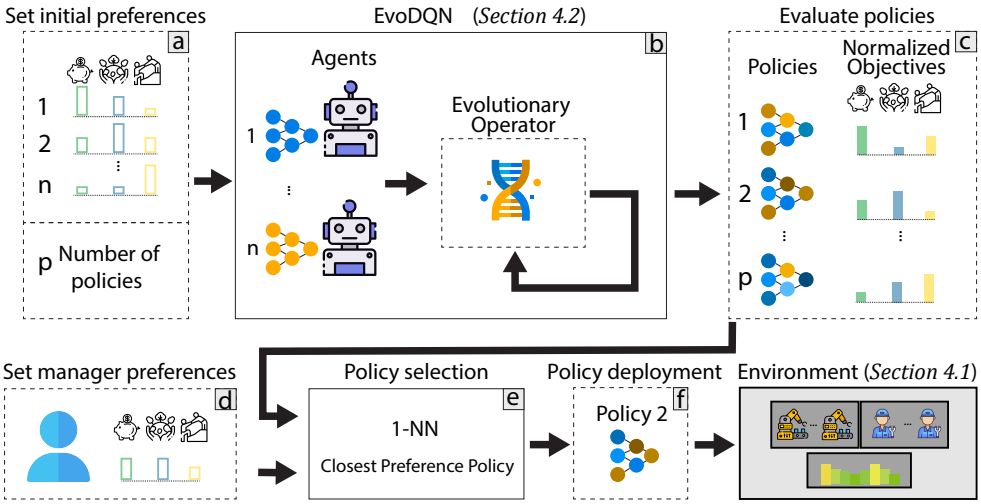


Figure 6.4: Overview of the EvoDQN framework

Section 6.3.1 introduces the problem and formalization underlying our DRL system and its interaction. Section 6.3.2 presents the EvoDQN algorithm, detailing the interactions agents have with the environment. All notations used in this work are summarized in Table 6.1.

6.3.1 Problem formulation and modeling

The formalization of the sustainable maintenance scheduling problem is carried out in an environment that involves a set of parallel machines that are repaired by technicians considering TBL. In this regard, section 6.3.1 describes the environment in which the maintenance scheduling problem exists, outlining the key components and interactions within the system. Following this, section 6.3.1 formalizes the problem within a DRL framework, capturing the decision-making process required to optimize multiple conflicting objectives simultaneously.

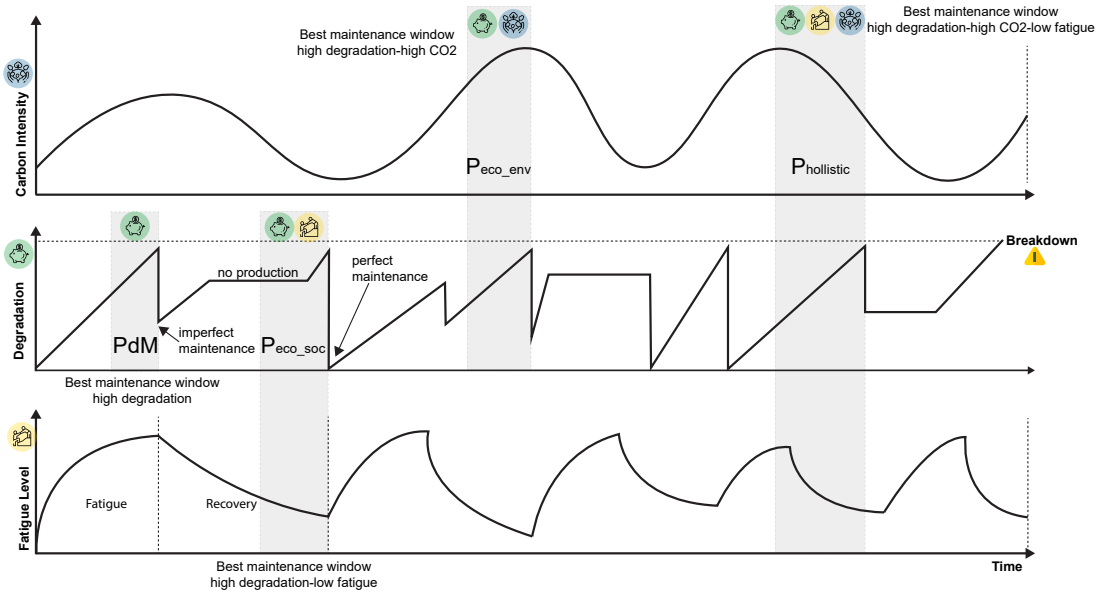


Figure 6.5: Illustration of the system signals and maintenance windows based on the TBL

System description

Let us consider a factory where the production is performed by a set of M parallel independent machines as shown in Figure 6.6. Each machine $m \in M$ suffers from a degradation modeled by a Gamma process Γ_m with shape parameter α_m and scale parameter β_m. The degradation reduces the efficiency of the production cycle linearly [Arı21], from full efficiency without any degradation to decreasing toward half efficiency. The production cycle stops completely when the machine is not operating (breakdown or maintenance). Machine $m \in M$ fails if the degradation reaches the threshold ρ_m. When machine m fails, a breakdown cost C^B is incurred, an additional downtime cost C^D is added for each time step in which the state of machine ω_m is in a breakdown ($\omega_m = 1$) or maintenance ($\omega_m = 2$). A machine m

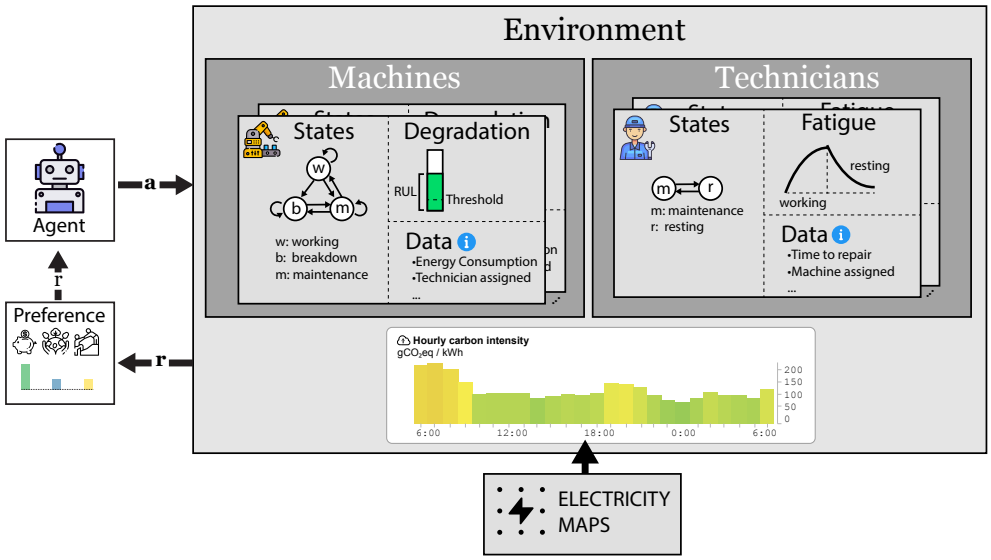


Figure 6.6: EvoDQN environment for sustainable maintenance policies generation

can receive maintenance by any available technician $t \in T$ to restore the state of the machine (imperfect maintenance action κ_i), or replacement (perfect maintenance κ_p) leaving the machine completely restored. When a technician performs maintenance ($\omega_t = 1$), the technician cannot be assigned to any other task until maintenance is completed, after which the technician is available again ($\omega_t = 0$). Each $t \in T$ has different maintenance times $\mathcal{T}_{tm}$ for each machine m, which somehow represents the technicians' skills. When a technician performs maintenance, the technician fatigue level increases. In contrast, when the technician is not assigned to any maintenance activity, he/she recovers from fatigue. The fatigue and recovery models described in [JGN13] are employed. These are detailed in (6.1) and (6.2), respectively.

$$F(\tau^F) = 1 - e^{-\lambda^F \tau^F} \tag{6.1}$$

$$\mathcal{R}(\tau^{\mathcal{R}}) = F(\tau) e^{-\mu^R \tau_i^{\mathcal{R}}} \tag{6.2}$$

The fatigue accumulated by time τ^F is represented by $F(\tau^F)$, while the residual fatigue after a rest period of length $\tau^{\mathcal{R}} \geq 0$ is given by $\mathcal{R}(\tau^{\mathcal{R}})$. The value of $\mathcal{R} = 0$ represents full recovery (no residual fatigue), while $\mathcal{R} = 1$ represents no recovery (maximum fatigue). In (6.1) and (6.2), λ^F and μ^R are fatigue and recovery parameters, respectively. These parameters control the speed of fatigue accumulation and recovery relief. This fatigue level increases the time it takes a technician to perform maintenance by $(1 + F(\tau^F))$. An example illustrating the different phases of fatigue and recovery for a technician is provided in Figure 6.5 based on a scenario of maintenance interventions over time.

In addition, an unexpected breakdown increases the total repair time by $\tau^B \sim \mathcal{N}(\mu_{\text{break}}, \sigma^2_{\text{break}})$. When a technician $t \in T$ performs maintenance, a maintenance cost is incurred depending on the type of maintenance action, C^P for perfect maintenance and C^I for imperfect maintenance.

Each machine m has an energy consumption (kWh) given by E_m. This energy consumption is a key factor in reducing the overall carbon footprint of the production process. Due to the fact that the carbon intensity, measured in (gCO2eq/kWh), fluctuates throughout the day, scheduling maintenance activities during periods of high carbon intensity can effectively reduce total carbon emissions associated with energy consumption. As an example, Figure 6.5 shows the different types of signals present in the system and the behavior of different maintenance policies that are adapted based on the TBL. For example, an economic maintenance policy (CM, PM, PdM) can focus on optimizing only the cost that takes into account the maintenance interventions based on the degradation signal (or time-based). A $\text{P}_{\text{eco\_soc}}$ can strike a balance between looking for the best maintenance window to reduce the cost and maintaining a low level of fatigue of the technicians to perform better maintenance interventions. A $\text{P}_{\text{eco\_env}}$ can decide to perform maintenance interventions to reduce costs while at the same time looking to do it when the carbon intensity is high. Finally, $\text{P}_{\text{eco\_hollistic}}$ can look for a balance between all these objectives. The primary objective is to obtain $\text{P}_{\text{eco\_hollistic}}$ policies that balance the costs associated with maintenance, machine breakdowns, and downtime with the goal of reducing the carbon footprint of the machines and the fatigue of the technicians.

Multi-Objective Markov Decision Process

To obtain optimal or near-optimal maintenance policies, it is necessary for decision-making agents to learn how to explore the preference space of multiple objectives in an environment whose dynamics are initially unknown [SB18]. This environment can be described by a Multi-Objective Markov Decision Process (MOMDP). MOMDP is a mathematical formulation of a problem in which an agent (decision-maker) selects actions sequentially to transit through different states guided by rewards. A MOMDP can be expressed as a 5-tuple $\langle S, A, \mathcal{P}, R, \gamma \rangle$, which consists of (i) a state space S indicating all possible states the agent can be in; an action space A indicating all possible actions the agent can take; (ii) a transition function $\mathcal{P} : S \times A \to S$ indicating the probability of transitioning from any state $s \in S$ to state $s' \in S$ given that the agent took action $a \in A$; (iii) a reward function $R : S \times A \times S \to \mathbb{R}^N$ that returns an immediate reward vector given by the transition from (s, a) to s'; and (iv) a discount factor $\gamma \in [0, 1]$ indicating how myopic the agent is, where $\gamma = 0$ indicates that the agent only cares about immediate reward and $\gamma \to 1$ indicates it gives more weight to future state information.

Table 6.2: The eight dispatching rules taken by the MOMDP agents

Rule		Description
1	$M_{HDL} \times T_{mF} \times \kappa_p$	Perform a perfect maintenance action by the technician with the lowest fatigue level on the machine with the highest degradation level.
2	$M_{HDL} \times T_{mF} \times \kappa_i$	Perform an imperfect maintenance action by the technician with the lowest fatigue level on the machine with the highest degradation level.
3	$M_{HDL} \times T_{MS} \times \kappa_p$	Perform a perfect maintenance action by the best performance technician on the machine with the highest degradation level.
4	$M_{HDL} \times T_{MS} \times \kappa_i$	Perform an imperfect maintenance action by the best performance technician on the machine with the highest degradation level.
5	$M_{HCT} \times T_{mF} \times \kappa_p$	Perform a perfect maintenance action by the technician with the lowest fatigue level on the machine with the highest cycle time.
6	$M_{HCT} \times T_{mF} \times \kappa_i$	Perform an imperfect maintenance action by the technician with the lowest fatigue level on the machine with the highest cycle time.
7	$M_{HCT} \times T_{MS} \times \kappa_p$	Perform a perfect maintenance action by the best performance technician on the machine with the highest cycle time.
8	$M_{HCT} \times T_{MS} \times \kappa_i$	Perform an imperfect maintenance action by the best performance technician with the lowest fatigue level on the machine with the highest cycle time.

The proposed sustainable maintenance policies aim to optimize three objectives: (i) minimize maintenance, breakdown and downtime costs; (ii) minimize technician fatigue level after performing maintenance actions; and (iii) minimize carbon footprint. Using the MOMDP framework, agents determine their actions based on the observations made by the system. Therefore, it is required to design local observations that provide the basis for the agent's actions and provide useful information for training.

The system state observed by the agent is represented by the vector given in (6.3), where $g_m^{\text{Norm}}\ \forall m \in M$ gives information about the normalized machine degradation, $q_t^{\text{Norm}}\ \forall t \in T$ is the normalized remaining time of each technician performing maintenance, η^{Norm} the normalized carbon footprint forecast, $\upsilon_{\mathbf{t}}\ \forall t \in T$ the fatigue level of the technicians, $O(\omega_m)\ \forall m \in M$ the one hot encoding of the state of each machine, and $O(\omega_t)\ \forall t \in T$ the one hot encoding of the state of each technician.

$$\mathbf{o} = (g_m^{\text{Norm}}, q_t^{\text{Norm}}, \eta^{\text{Norm}}, \upsilon_{\mathbf{t}}, O(\omega_m), O(\omega_t)) \tag{6.3}$$

Depending on the observation, the agent decides at each time step which type of maintenance operation needs to be performed. Eight rules are defined based on the selection of the machine, technician, and type of maintenance, as detailed hereinafter:

- **Machine selection** based on Highest Degradation Level (m_{HDL}), and Highest Cycle Time (m_{HCT})
- **Technician selection** based on Technician with the minimum level of fatigue (t_{mF}) or maximum skills (t_{MS}).
- **Maintenance operation** based on the type of maintenance, perfect maintenance (κ_p), where the component is replaced, or imperfect maintenance (κ_i) where the state of a component is partially recovered.

The action taken by the agent is given in (6.4), in addition, the policy can decide not to take any action (no technician will be assigned to perform maintenance) represented by d. The eight dispatching rules are detailed and explained in Table 6.2.

$$\mathbf{a}_m \in \{M_{HDL}, M_{HCT}\} \times \{T_{mF}, T_{MS}\} \times \{\kappa_p, \kappa_i\} \cup \{d\} \tag{6.4}$$

To guide agents in learning the policy, a reward vector is defined to represent the three pilars of sustainability, as given in (6.5).

$$\mathbf{r} = [r_{eco}, r_{env}, r_{soc}] \tag{6.5}$$

The economic pillar, represented by r_{eco}, is integrated by three elements. r^1_{eco} is the cost of performing a perfect or imperfect maintenance operation by a given technician and machine. r^2_{eco} is the downtime cost for machines that are not in an operational state (breakdown or maintenance). r^3_{eco} is the breakdown cost when a machine goes into a breakdown state. The following equations represent each component of the reward function of the economic pillar:

$$r^1_{eco}(s, a, s') = (C^P \mathbb{1}_{\kappa_p \in a} + C^I \mathbb{1}_{\kappa_i \in a}) \forall m \in M \tag{6.6}$$

$$r^2_{eco}(s, a, s') = \sum C^D \mathbb{1}_{\omega_m > 0} \forall m \in M \tag{6.7}$$

$$r^3_{eco}(s, a, s') = \sum_{m \in M} C^B \mathbb{1}_{\omega_m = 0 \wedge \omega'_m = 2} \tag{6.8}$$

$$r_{eco} = -\frac{(r^1_{eco} + r^2_{eco} + r^3_{eco})}{\text{max possible cost}} \tag{6.9}$$

The environmental pillar, represented by r_{env}, indicates the carbon footprint of the system, which is calculated as follows:

$$r_{env} = -\frac{\eta_0 \sum_{m \in M} E_m \mathbb{1}_{\omega_m = 0}}{\text{max energy consumption}}$$

The social pillar, represented by r_{soc}, is defined as the sum of the fatigue of the technicians when performing maintenance interventions and the sum of the absolute difference between the fatigue of the technicians, thus promoting a balanced workload (weighted by ξ) while minimizing fatigue.

$$r_{soc} = -\left(\xi \frac{\sum \upsilon_i}{|T|} + (1 - \xi) \frac{\sum_{i \neq j} |\upsilon_i - \upsilon_j|}{2|T|}\right) \forall i, j \in T$$

The transition function $\mathcal{P}$ captures the dynamics of the system, including the stochastic degradation of machines, CO2 forecast, and the relationship between production and maintenance. For a particular action a, given a state s, the next state s' is determined as follows:

- Machine degradation following a Γ process and the degradation is only restored by maintenance actions κ_p and κ_i. A perfect maintenance action restores the machine completely. An imperfect maintenance action restores partially the original state of the machine.
- Remaining technician time decreases for each timestep and is normalized by the maximum repair time.
- Forecast of CO2 is determined by the hourly data of the Carbon Intensity of Luxembourg from ElectricityMaps [Ele24]. Taking the values from the current step x to $x+\eta^s$, the signal is added to a noise vector that is generated using a linearly spaced vector that goes from 0 to 1 over the η^s forecast steps, scaled by random values drawn from a normal distribution with μ_{car} and σ_{car} parameters.
- Fatigue level of each technician increases on maintenance interventions while decreases on the resting periods following the fatigue and recovery models formalized in (6.1) and (6.2).

Overall, the objective is to find policies that maximize the hypervolume.

6.3.2 EvoDQN algorithm

Combining evolutionary algorithms with AI, such as DRL, has been successful in many applications [Dru19], notably for solving dynamic scheduling problems [CYL+20; KLV+22; SWY+23; FGC+22]. Based on this evidence, this research develops a novel Evolutionary multi-objective multi-agent DQN (EvoDQN), detailed in Algorithm 4, to generate sustainable policies that balance the different objectives of the TBL. DRL is responsible for optimizing the policies, while evolutionary computation generates new policies in the preference space, the goal being to maximize the hypervolume of these policies by training a limited set of policies in each generation.

In a more technical way, EvoDQN distributes and optimizes multiple agents across different initial preferences, as was sketched schematically in Figure 6.4. The initial set of policies then undergoes an evolutionary process in which new agents are generated through an evolutionary operator with the goal of maximizing hypervolume. According to the methodology proposed by [BDL20], EvoDQN combines agents that are closely aligned in terms of preferences. These agents are trained based on DQN [MKS+13], where the reward vector is scalarized based on the preference vector. Once the agents have been trained, the selection of agents is based on two functions. The first is `Pareto_Agents`, which selects agents that form a Pareto front in the reward space based on the cumulative reward. The second

Algorithm 4 Evolutionary multi-objective multi-agent Deep Q-Network (EvoDQN)

Input: *num_generations*, *initial_preferences*, *num_steps*, *num_policies*, *mutation_rate*, *crossover_rate*
for each i **in** *initial_preferences* **do**
    Initialize agent i and store in A
    Initialize replay memory D_i
    Initialize action-value function Q_{θ_i} with random weights θ_i
    Initialize target action-value function $Q_{\theta'_i}$ with weights $\theta'_i = \theta_i$
    Set the preference vector $\mathbf{d}_i$ for agent i
for $g \leftarrow 1$ **to** *num_generations* **do**
    for each agent i **in** A **do**
        $a_j \leftarrow$ closest neighbor of a_i by preference vector
        if $g > 1$ **then**
            $a_i^*, a_j^* \leftarrow$ Evolutionary_Operator$(a_i, a_j, mutation_rate, crossover_rate)$
            $A \leftarrow A \cup \{a_i^*, a_j^*\}$
    for each agent i **in** A **do**
        for *step* $\leftarrow 1$ **to** *num_steps* **do**
            Choose a from s using an ϵ-greedy approach
            Take action a, observe reward $\mathbf{r}$ and next state s'
            $r' \leftarrow \mathbf{r} \cdot \mathbf{d}$
            Store transition (s, a, r', s') in D_i
            Sample random minibatch of transitions (s_j, a_j, r'_j, s'_j) from D_i
            Set

$$y_j = \begin{cases} r'_j & \text{for terminal } s' \\ r'_j + \gamma \max_{a'} Q_{\theta'}(s'_j, a') & \text{otherwise} \end{cases}$$

            Perform a gradient descent step on $(y_j - Q_\theta(s_j, a_j))$ with respect to θ
            $s \leftarrow s'$
    $A \leftarrow$ Pareto_Agents(A)
    $A \leftarrow$ Hypervolumen_Contrib(A, *num_policies*)

is `Hypervolume_Contrib`, which selects the *num_policies* that contributes most to maximizing hypervolume. The evolutionary operator, detailed in Algorithm 5, is applied to each agent with its nearest neighbor in the preference space. The objective is to combine the experiences of pairs of agents that are close to each other in their preference space, while perturbing the weights of the policy network and moving the preference vector towards each other to create two new pairs of agents. To ease understanding, an overview of how the evolutionary operator is presented in Figure 6.7.

6.4 Implementation & Evaluation

In the following, EvoDQN is compared with classical maintenance policies. Section 6.4.1 introduces the experimental setup used for evaluation. Section 6.4.2 details the methodology employed to train EvoDQN and their baseline (cf. Fig-

Algorithm 5 Evolutionary_Operator

Input: a_1 /*Agent 1*/, a_2 /*Agent 2*/
$a_1^* \leftarrow a_1$
$a_2^* \leftarrow a_2$
▷ Crossover through the replay buffer
Let N be the length of the replay buffer
$rb\_co \sim \mathcal{U}(0.05, 0.95)$
$co\_idx \leftarrow 1 - crossover\_rate \cdot rb\_co$
$rb_{11} \leftarrow a_1.replay\_buffer(0, co\_idx)$
$rb_{21} \leftarrow a_2.replay\_buffer(0, co\_idx)$
$rb_{12} \leftarrow a_1.replay\_buffer(co\_idx, N)$
$rb_{22} \leftarrow a_2.replay\_buffer(co\_idx, N)$
$a_1^*.replay\_buffer \leftarrow \text{merge}(rb_{11}, rb_{22})$
$a_2^*.replay\_buffer \leftarrow \text{merge}(rb_{21}, rb_{12})$
▷ Mutation through the weights and preferences
$pv\_mu \sim \mathcal{U}(0.05, 0.95)$
$mr \leftarrow 1 - mutation\_rate \cdot pv\_mu$
$a_1^*.pref = (mr) \cdot a_1.pref + (1 - mr) \cdot a_2.pref$
$a_2^*.pref = (1 - mr) \cdot a_1.pref + mr \cdot a_2.pref$
$\epsilon_1, \epsilon_2 \sim \mathcal{N}(\mu, \sigma^2)$
$a_1^*.policy\_weights = a_1.policy\_weights + \epsilon_1$
$a_1^*.target\_weights \leftarrow a_1.policy\_weights$
$a_2^*.policy\_weights = a_2.policy\_weights + \epsilon_2$
$a_2^*.target\_weights \leftarrow a_2.policy\_weights$
return a_1^*, a_2^*

ure 6.4 to visualize where this step stands). The results obtained between the different policies and their sustainable implications are discussed in section 6.4.3.

6.4.1 Evaluation & environment setup

The evaluation was conducted in a three-machine, two-technician setting. Each machine has an energy consumption of $E_m = 100.0$ kWh $\forall m \in M$. The gamma degradation model is defined with a shape parameter of $\alpha_m = 2.0$ and a scale parameter of $\beta_m = 1.0$ on all machines. The failure threshold is set to $\rho_m = 20.0$ $\forall m \in M$. The maintenance time of the two technicians are defined as $\mathcal{T}_{t=1,m} = 6$ and $\mathcal{T}_{t=2,m} = 3$ $\forall m \in M$ units of time to complete any intervention in a complete rest state. The parameters for the fatigue and recovery model presented in 6.1 and 6.2 are $\lambda^F = 0.03$ and $\mu^R = .15$. The cost of perfect maintenance is $C^P = 500.0$ units per machine, while imperfect maintenance costs $C^I = 100.0$ units per machine. The downtime cost is $C^D = 1.0k$ units and the cost of a machine breakdown is significantly higher at $C^B = 10.0k$ units. We obtained information on the carbon intensity in Luxembourg in 2022 from the Data Portal of Electricy Maps. The carbon intensity represents the amount of greenhouse gases emitted per unit of electricity consumed as grams of CO2 equivalent per kilowatt hour

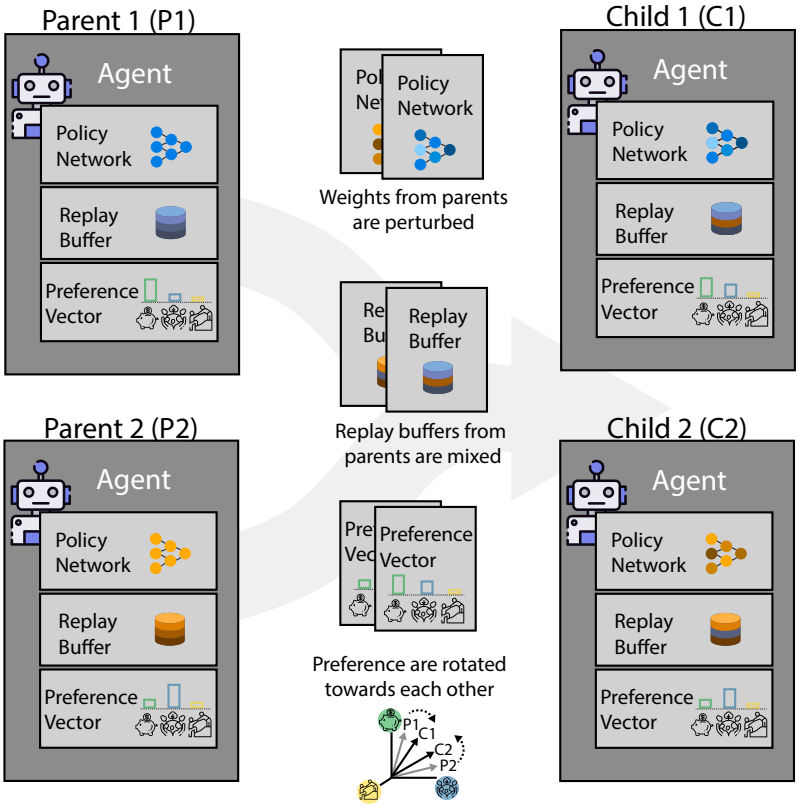


Figure 6.7: Overview of the Evolutionary Operator

(gCO2eq/kWh). From the carbon intensity signal, $\eta^s = 10$ for the forecast with a noise vector using $\mu_{\text{car}} = 0.0$ y $\sigma_{\text{car}} = 0.1$. A summary of the overall experimental configuration is provided in Table 6.3.

The evaluation of maintenance policies is based on three pillars of sustainability: economic, social, and environmental. The economic pillar considers maintenance costs, breakdown costs, and downtime costs. The social pillar assesses the fatigue level of technicians performing maintenance. The environmental pillar measures the carbon footprint on the production of machines.

The policies obtained by EvoDQN are compared against two classical maintenance policies: CM, and CBM. These policies are described in the following.

- The CM policy performs perfect maintenance whenever the machine suffers a degradation that causes the equipment to go into a breakdown state. As half of the defined dispatching rules perform perfect maintenance (Rule 1,3,5 7 in Table 6.2), the CM policy only executes those rules;
- CBM policy performs maintenance activities by executing one of the eight dispatching rules when a machine reaches ρ_m.

Table 6.3: System evaluation parameters

Category	Parameter
Machine Parameters	
Number of machines	3
Energy consumption/m (kWh)	100.0
Gamma shape parameter	2.0
Gamma scale parameter	1.0
Failure threshold	20.0
Technician Parameters	
Number of technicians	2
Maintenance time for T1	6
Maintenance time for T2	3
Workload balance	0.5
Cost Parameters	
Perfect maintenance/m	0.5k
Imperfect maintenance/m	0.1k
Downtime	1.0k
Breakdown	10.0k
Forecast CO2 Parameters	
Steps	10
Mean noise	0.0
Std noise	0.1

CM only executes four dispatching rules since it needs to perform perfect maintenance (replacement) when the component breaks. On the other hand, CBM executes a dispatching rule when a particular threshold is reached on a machine. We have evaluates 99 different thresholds for CBM (from 1 to 99). In total, 8 policies were obtained for CM and 792 policies for CBM (99 thresholds $\times$ 8 rules).

6.4.2 Training of evolutionary DRL agents

For EvoDQN training purposes, the Wandb Sweep library is used to perform a Bayesian search for hyperparameters that maximize the hypervolume. A total of 40 parallel sweep agents were executed with a maximum of 48 hours running on 2.6 GHz AMD Epyc ROME 7H12 nodes. A total of 144 EvoDQN models were trained. Each agent on the evolutionary approach was trained for 250k steps (e.g., if there are 10 agents in the first generation, a total of 2.5M steps will be taken up to that point) and the different objectives are validated on 50 episodes. The set of initial preferences is the one reported in Table 6.4. Each of the agents receives one of these preferences that scalarizes the reward vector while being trained (cf., Algorithm 4). Following this, new agents are created through the evolutionary process (cf., Algorithm 5). The configuration of the sweep parameters and the best

Table 6.4: Matrix of initial preferences

	Weights		
	Eco	**Env**	**Soc**
Preference 1	0.25	0.5	0.25
Preference 2	0.00	0.75	0.25
Preference 3	0.00	1.00	0.00
Preference 4	0.25	0.00	0.75
Preference 5	0.00	0.25	0.75
Preference 6	1.00	0.00	0.00
Preference 7	0.75	0.00	0.25
Preference 8	0.50	0.25	0.25
Preference 9	0.25	0.75	0.00
Preference 10	0.25	0.25	0.50
Preference 11	0.75	0.25	0.00
Preference 12	0.50	0.50	0.00
Preference 13	0.00	0.00	1.00
Preference 14	0.00	0.50	0.50
Preference 15	0.50	0.00	0.50

parameters obtained are detailed in Table 6.5, where the best parameters were those that maximized the hypervolume with a maximum selection of eight agents across generations.

After obtaining the parameters that maximize hypervolume, nine more models are trained to compare our approach with, as given in Figure 6.8. The different models are described hereinafter:

- **EvoDQN**: EvoDQN employs a selection process of 8 agents per generation, each trained for $250k$ steps using the initial preferences from Table 6.4. Total number of steps taken are $42.25M$;
- **EvoDQN***: EvoDQN employs a selection process of 8 agents per generation, each trained for $250k$ steps using the initial preferences of the last generation of EvoDQN. Total number of steps taken was $5.5M$;
- **DQN_I**: DQN with 15 agents, each being trained for $\approx 2.816M$ steps with the preferences obtained by the last generation of EvoDQN. Total number of steps taken are $42.25M$;
- **DQN_F**: DQN with 8 agents, each being trained for $\approx 5.281M$ steps with the preferences obtained by the last generation of EvoDQN. Total number of steps taken are $42.25M$ among all agents;
- **DQN_I250k**: DQN with 15 agents, each being trained for $250k$ steps using the initial preferences shown in Table 6.4. A total of $3.75M$ steps were taken.
- **DQN_F250k**: DQN with 8 policies, each being trained for $250k$ steps with the preferences obtained by the last generation of EvoDQN. number of steps

Table 6.5: Hyperparameter sweep configuration & best values

Parameter	**Values**	
Name	**Sweep**	**Best**
Batch Size	16, 32, 64, 256	256
Buffer Size	5k, 10k, 20k	5k
Layer Size	64, 256, 1024, 2048	256
Learning Rate End	0.0001 to 0.001	0.001
Learning Rate Start	0.001 to 0.1	0.080
Mutation rate	0.05 to 0.5	0.303
Number of Gen.	5, 10, 15, 20	20
Perturbation rate	1e-05 to 0.1	0.057
Target Steps	32, 64, 128, 256	64
Tau	0.01 to 0.1	0.098
Update Steps	4, 8, 16, 32	32

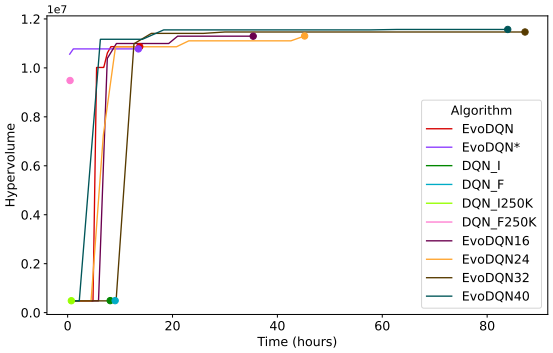


Figure 6.8: Hypervolume over time for EvoDQN and DQN variants

taken are $2M$ among all agents;

- **EvoDQN16**: EvoDQN with 16 agents, each being trained for $250k$ steps with initial preferences from Table 6.4;
- **EvoDQN24**: EvoDQN with 24 agents, each being trained for $250k$ steps with initial preferences from Table 6.4.
- **EvoDQN32**: EvoDQN with 32 agents, each being trained for $250k$ steps with initial preferences from Table 6.4.
- **EvoDQN40**: EvoDQN with 40 agents, each being trained for $250k$ steps with initial preferences from Table 6.4.

EvoDQN is compared with DQN-based models, DQN_I and DQN_I250K, with the initial preferences from Table 6.4. In addition, the final preferences of the EvoDQN agents, obtained by the evolutionary processes, were used as initial

preferences for the DQN-based models DQN_F and DQN_F250K. In addition, EvoDQN* – *which is EvoDQN but with the initial preferences as the ones obtained by EvoDQN* – is trained and evaluated. Furthermore, we analysed the performance of EvoDQN with an increased number of agents (16, 24, 32 and 40) using the initial preferences outlined in Table 6.4. The models were named EvoDQN16, EvoDQN24, EvoDQN32 and EvoDQN40, respectively.

After analyzing the different models, as show in Figure 6.8, EvoDQN found policies that are better distributed in the preference space and obtain a higher hypervolume compared to DQN-based. It is noteworthy to say that the preferences assigned to DQN do not vary in training, so we train the models of DQN_I and DQN_I250k with the initial preferences in Table 6.4, and DQN_F and DQN_F250K with the preferences found by EvoDQN. The policies found by DQN_F, although trained with the same number of total steps as EvoDQN, overfit and give higher priority to environmental and social objectives than to economic ones. The DQNF250K policies demonstrated an improvement in hypervolume, although the extent of this improvement was less pronounced than that observed with EvoDQN and EvoDQN*. When comparing the performance of EvoDQN and EvoDQN*, it is evident that a wider field of initial preferences allows for a more comprehensive exploration, rather than focusing on the region where the hypervolume is maximized by EvoDQN. This approach leads to a higher hypervolume than that achieved by EvoDQN*. Moreover, increasing the number of agents involved in the selection process leads to an increase in the hypervolume, but also a longer training time, due to the fact that more agents are trained per generation.

6.4.3 Maintenance policy comparison analysis

As previously mentioned in section 6.4.1, EvoDQN with eight policies is compared with the CBM and CM policies, where each policy represents a different dispatching rule. We selected CBM with a threshold of 77 (denoted by CBM77) as the set of policies to evaluate because it demonstrated the highest level of effectiveness compared to the other thresholds in terms of hypervolume. Although these policies are primarily focused on the economic aspect, we are interested in evaluating the impact on the other pillars. Before comparing those policies, we believe that it is important to provide insight into what one EvoDQN policy implies on the system and technician behaviors over a period of time. In this regard, section 6.4.3 showcases the sustainable implication for policy 7 (P7) of EvoDQN. Thereafter, the comparison results are discussed in section 6.4.3

Sustainable implication of EvoDQN P7

Economic policies are effective due to the straightforward relationship between profit maximization and cost minimization. However, the inclusion of sustainable

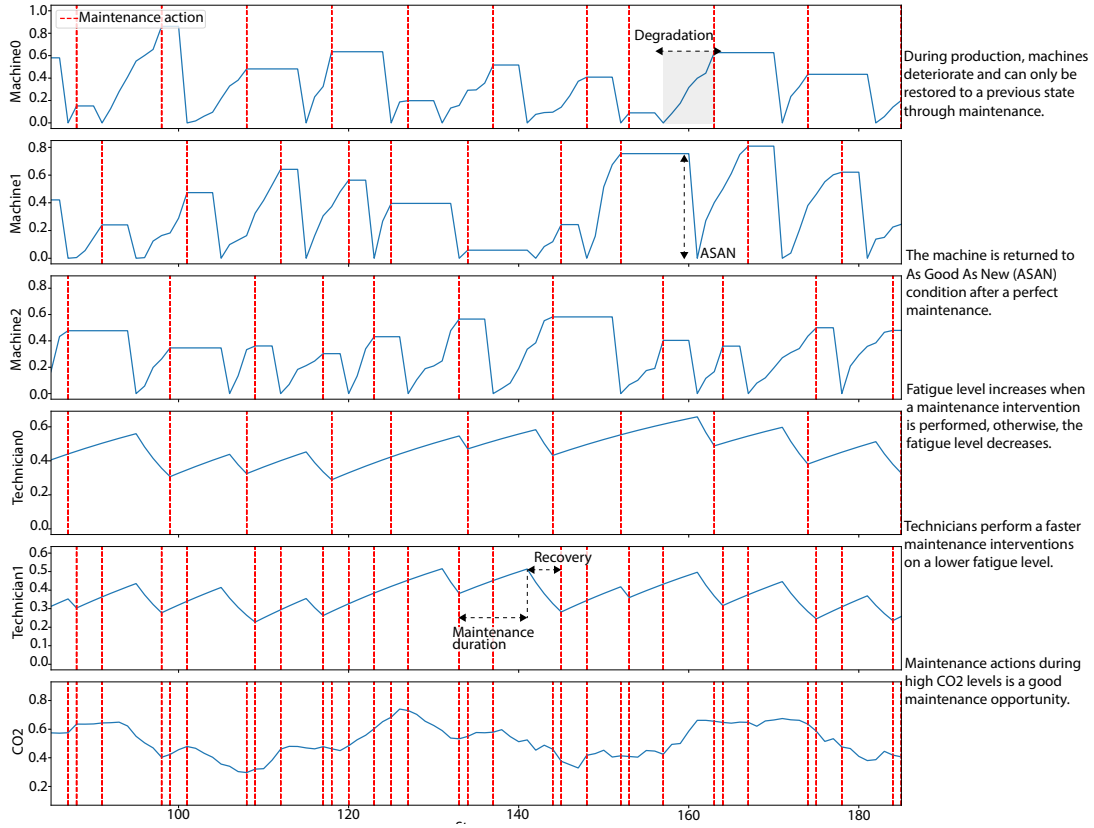


Figure 6.9: Machine degradation, technician fatigue, and carbon intensity associated with maintenance interventions

policies directly impacts the economic objective. For example, identifying windows of opportunity where the carbon footprint is high to perform maintenance interventions may benefit the environment, but it may also affect production and maintenance by performing interventions far from the point of failure (too early or too late). Similarly, reducing fatigue and distributing maintenance tasks can affect the selection of technicians and the timing of the intervention. In practice, only a subset of policies that cover the entire preference space is useful. For example, policies that prioritize sustainability over economic considerations may only perform maintenance when the carbon footprint is high. Extreme cases in the preference space may prevent machines from continuing in production to minimize the carbon footprint. On the other hand, the suggested socially beneficial policies would prevent technicians from experiencing high levels of fatigue by prioritizing rest over interventions. In the most extreme cases, technicians would never perform maintenance activities to avoid increasing fatigue. Figure 6.9 presents a subset of the temporal data of a policy where the three objectives are balanced. The first

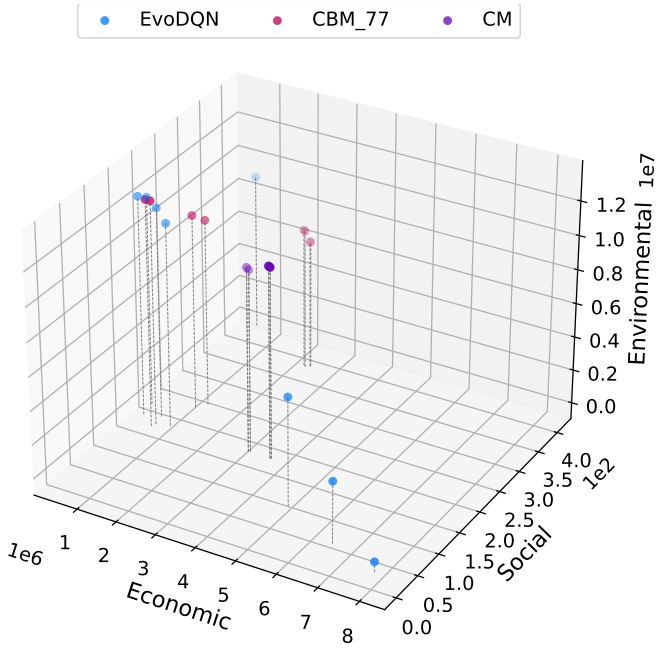


Figure 6.10: EvoDQN, CBM77 and CM policies for the TBL

three temporal data graphs (starting at the top) represent the degradation of the machines where the breakdown point is when the degradation reaches 1.0. The following two graphs illustrate the fatigue level of the two technicians involved in the maintenance process, the value of 0 and 1 representing respectively complete recovery and fatigue (cf., Equations (6.1) and (6.2) for further details). Finally, the last graph (at the bottom) represents the carbon footprint signal of the overall manufacturing process.

Comparison results

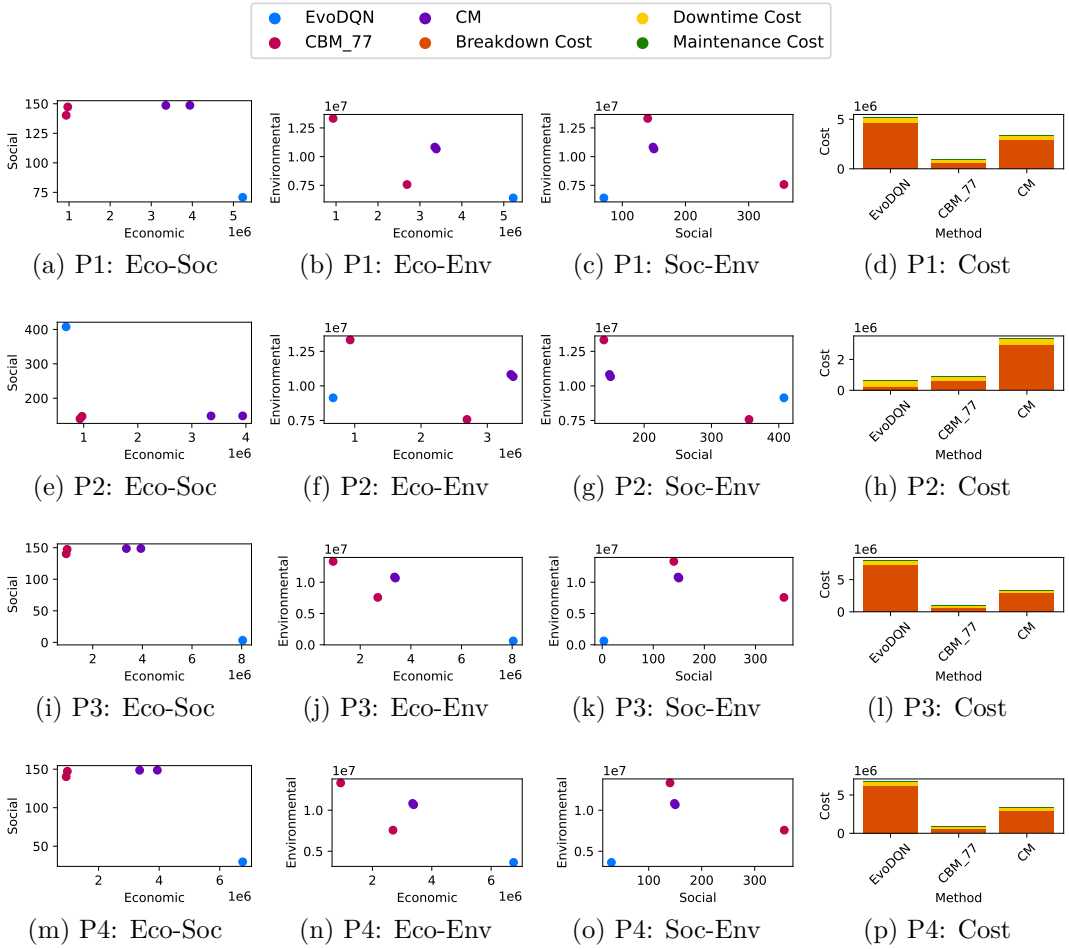
The results in terms of the trade-off between economic, social, and environmental objectives for the maintenance policies of EvoDQN, CBM77 and CM are presented in Figure 6.10. The lower the score on all the axes, the better the policy. It can be observed that the EvoDQN policies clearly illustrate a trade-off between the economic, environmental, and social objectives. It is evident that as social and environmental costs increase, the economic cost decreases. In particular, we can observe a set of policies in the lower part of the economic axis that exhibit a slight trade-off between the other two objectives. From the entire set of EvoDQN policies, a subset has been identified as feasible for implementation in an industrial setting. This is because policies that significantly reduce carbon footprint values and technician fatigue tend to avoid maintenance activities to minimize fatigue, or even, in the most extreme cases, to keep machines in a state of downtime (for

maintenance or breakdown) so that the carbon footprint is minimized. This kind of policies are, of course, not a viable option in real-world industrial scenarios. In the case of CBM77, certain policies demonstrate distinctive behavior, some exhibiting competitive economic results comparable to those of the EvoDQN policies. In contrast, CM policies exhibit a consistent, centralized cluster with similar behavior.

Table 6.6: Performance metrics for various policies

Policy	**Total Cost**		**Fatigue**		**CO2**	
	Mean	**Std**	**Mean**	**Std**	**Mean**	**Std**
EvoDQN8 P1	5224632.00	643776.33	70.81	23.05	6400227.78	1170575.91
EvoDQN8 P2	673694.00	193507.49	407.64	7.22	9135870.62	112714.30
EvoDQN8 P3	8041360.00	115930.81	3.28	2.86	611974.08	136320.13
EvoDQN8 P4	6756262.00	294488.76	29.71	6.53	3640282.90	896390.78
EvoDQN8 P5	906400.00	417111.25	165.02	7.83	12330807.18	503133.79
EvoDQN8 P6	1291790.00	574877.80	152.20	9.33	11953001.80	1107733.05
EvoDQN8 P7	434270.00	94808.87	183.53	8.60	12283418.08	261931.21
EvoDQN8 P8	513110.00	182068.51	158.62	7.41	12880761.30	319704.17
CBM77 R1	969090.00	220347.24	147.33	15.26	13124996.92	527897.96
CBM77 R2	2547168.00	214518.61	352.91	19.01	8248932.42	328836.38
CBM77 R3	930810.00	204269.26	140.21	14.04	13326043.14	485182.85
CBM77 R4	2600908.00	197973.47	350.42	20.71	8246078.68	285789.40
CBM77 R5	1365580.00	275681.47	202.72	21.43	11413359.98	686179.06
CBM77 R6	2692006.00	202550.53	354.06	22.65	7586123.62	404762.00
CBM77 R7	1543310.00	289439.75	215.63	30.50	10988890.86	908023.59
CBM77 R8	2693018.00	251266.04	355.77	18.95	7570451.52	380278.95
CM R1	3940120.00	172669.90	148.68	13.83	11208915.06	373082.92
CM R2	3901730.00	172146.63	148.85	14.18	11253092.26	436529.93
CM R3	3391030.00	207082.07	150.18	12.11	10667375.66	486925.62
CM R4	3355220.00	238993.52	148.59	17.59	10827049.70	561767.31

For deeper analysis, the results obtained from the eight EvoDQN policies, eight CBM77, and four CM policies are presented with their 2D projection in Figure 6.11 and Figure 6.12, along with a more in-depth breakdown of the economic pillar. The first three columns represent the projection of the trade-off between two objectives, the first column representing Economic-Social, the second column representing Economic-Environmental, and the third one representing Social-Environmental. As the CM and CBM policies are rule-based, which EvoDQN is able to select depending on the state of the system, we present the two best rules of the CM and CBM from the different projections in the first three columns and the best result of each policy for the last column. On the EvoDQN side, we present in each row the eight different policies found. The fourth column provides in the form of a bar chart the different costs that conforms the Economic pillar. From an economic standpoint, policies P2, P7, and P8 demonstrate superior performance



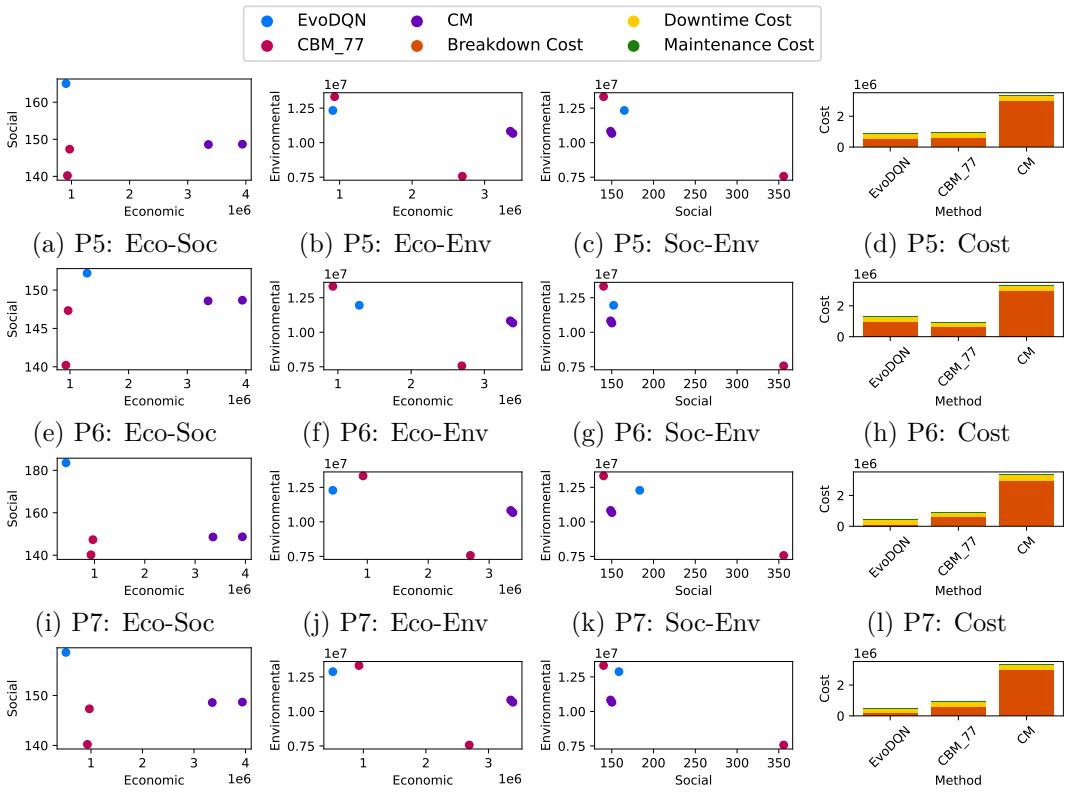
(a) P1: Eco-Soc (b) P1: Eco-Env (c) P1: Soc-Env (d) P1: Cost

(e) P2: Eco-Soc (f) P2: Eco-Env (g) P2: Soc-Env (h) P2: Cost

(i) P3: Eco-Soc (j) P3: Eco-Env (k) P3: Soc-Env (l) P3: Cost

(m) P4: Eco-Soc (n) P4: Eco-Env (o) P4: Soc-Env (p) P4: Cost

Figure 6.11: 2D Projection of the policies P1 to P4 in the objective space and economic cost breakdown



(a) P5: Eco-Soc (b) P5: Eco-Env (c) P5: Soc-Env (d) P5: Cost

(e) P6: Eco-Soc (f) P6: Eco-Env (g) P6: Soc-Env (h) P6: Cost

(i) P7: Eco-Soc (j) P7: Eco-Env (k) P7: Soc-Env (l) P7: Cost

(m) P8: Eco-Soc (n) P8: Eco-Env (o) P8: Soc-Env (p) 8: Cost

Figure 6.12: 2D Projection of the policies P5 to P8 in the objective space and economic cost breakdown

Table 6.7: Production cycle mean and standard deviation

Policy	PC Mean	PC Std
EvoDQN8 P1	23182.70	1110.41
EvoDQN8 P2	31118.56	508.18
EvoDQN8 P3	19386.97	143.28
EvoDQN8 P4	20920.41	368.69
EvoDQN8 P5	29852.60	658.53
EvoDQN8 P6	28485.00	1183.37
EvoDQN8 P7	**31286.72**	**327.55**
EvoDQN8 P8	30850.90	437.16
CBM77 R1	28863.94	269.56
CBM77 R2	24800.80	324.57
CBM77 R3	28860.07	249.15
CBM77 R4	24740.38	329.07
CBM77 R5	**29074.47**	**287.84**
CBM77 R6	25796.58	576.20
CBM77 R7	28996.30	317.74
CBM77 R8	25749.26	511.02
CM R1	24238.93	176.51
CM R3	24286.44	183.50
CM R5	25561.90	382.69
CM R7	**25572.46**	**412.26**

compared to traditional maintenance policies, and policy P5 exhibits comparable results. CM policies are associated with higher breakdown costs, which represent the primary maintenance expense. It is clear that the CM policy incurs higher breakdown costs due to its reactive approach, in which maintenance action is only performed once a breakdown has occurred. In contrast, the CBM77 policy can prevent more breakdowns by performing maintenance before they occur, but it is challenging to coordinate the technicians involved in the process. In terms of social impact, policies P1, P3, and P4 have demonstrated the most effective approach to reducing fatigue. Furthermore, policies P5 to P8 have demonstrated intermediate solutions, with the best CM and CBM77 solutions being superior. In terms of reducing carbon footprint, policies P1, P3, and P4 are the most effective, in line with policies that prioritize social aspects. However, there are intermediate policies, such as policy P2, which significantly minimize the carbon footprint compared to the CM policies. With the exception of one CBM77 policy, they also optimize maintenance costs to a large extent.

6.4.4 Operational feasibility in industrial operations

It is essential for companies to achieve a balance between the dual objectives of maximizing profits and minimizing costs, while maintaining a sustainable approach.

However, policies that give more priority to environmental or social criteria than economic ones are not relevant in an industrial context. In this regard, the policies obtained by EvoDQN P2, P5, P7, and P8 are well-suited for implementation in industrial contexts where economic considerations are of primary importance. With respect to the pair of objectives, policy P8 shows notable performance in balancing social and economic considerations, while policy P2 exhibits a particularly strong economic and environmental focus. It is important to note that while some EvoDQN policies may not be directly relevant for implementation in operational settings, as previously discussed, they remain essential for an optimal exploration of the search space.

Although the CBM and CM policies are mainly oriented to economics (they do not take well into account the balance between the three TBL criteria), we can still highlight that the policies are viable options for application in real industrial-settings, especially when using the dispatch rule R1 and R3, which allows a good performance in the economic aspect and also reflected in the social aspect.

From an operational condition perspective, an important KPI to be analyzed is the production cycle of each policy. This KPI is summarized in Table 6.7 for all the 20 evaluated policies. The best results for each family of policies are highlighted in bold. The set of P2, P5, P7, and P8 policies demonstrate better performance compared to the traditional policies. P7 exhibits the most notable improvement in terms of the average production cycle, with a 8.40% increase compared to CBM77 and a 22.34% improvement for CM in its most cost-effective policies. These policies also represent the most cost-effective options, as the highest costs are, respectively, the breakdown and downtime, which represents production equipment with a higher production cycle.

6.5 Conclusion

This chapter presents a framework for developing sustainable maintenance policies based on a hybrid approach that combines evolutionary computation and DRL. Our proposal involves multiple agents, each with a preference vector that determines the reward. After training, the agents undergo an evolutionary process in which new agents are created and trained with the goal of maximizing the hypervolume and identifying a set of distributed policies within the preference space. In addition, a scenario based on parallel machines that experience degradation and repair by technicians was proposed, which incorporates the three pillars of sustainability. The proposed EvoDQN approach obtains distributed policies in the preference space demonstrating different trade-offs among the TBL objectives. The results showed how the economic pillar contrasts with the proposed social and environmental pillars. In addition, we found that our approach produces superior results in terms of the production cycle compared to classical maintenance policies

that lead to higher profits.

6.5.1 Implications

Maintenance optimization is a process that involves understanding the condition of the equipment by incorporating the use of technology to monitor variables in real time such as temperature, vibration, pressure, and humidity. The collection and analysis of this data using ML algorithms is essential for identifying patterns, correlations, and/or abnormalities that could indicate potential equipment failures or performance issues. However, it is important to consider the additional costs associated with implementing this technology, including the installation of sensors, network infrastructure, and computational power to train the ML models. Furthermore, incorporating social aspects can improve well-being and job satisfaction, as well as help identify specific training needs, enabling the development of new skills and the improvement of existing ones.

7

Conclusion

This chapter presents the overall conclusion of the dissertation and proposes potential research directions.

Contents

7.1 Summary of contributions

This thesis examines the application of reinforcement learning for the optimization of maintenance activities in the manufacturing industry, with a focus on the concepts of uncertainty, efficiency, and sustainability. The approach aligns with the optimization of maintenance processes through the concept of policy, which centers on the decision-making process regarding the assignment of maintenance tasks to technicians. The objective of this decision is to guarantee the continuity and efficiency of operations while minimizing downtime and costs. During this work we address different challenges.

In Chapter 4, we focused on investigating the impact of different levels of uncertainty on the distribution of failures and repair time, with the goal of optimizing machine uptime. Three different methods based on scheduling rules, GA (with multiple re-optimization frequencies), and RL were evaluated. Our results showed that RL showed a significant reduction in MTTR compared to the other methods with different levels of uncertainty; also, when using real data, it showed a high ability to maximize the uptime of the equipment. On the other hand, GA with high re-optimization frequency proved to be the best alternative in optimizing equipment uptime under different degrees of uncertainty, but it is computationally inefficient. Finally, DR are a simple alternative with low computational cost and competitive results in reducing the MTTR.

In Chapter 5, we investigated the use of reinforcement learning to design maintenance policies. In this work, classical maintenance policies were evaluated (i.e. CM and PM), where different metrics were analyzed to determine which policies are more effective in terms of number of actions, number of failures, percentage of different states of the equipment (in maintenance, operational, breakdown), where the objective is to perform the maintenance actions as close as possible to the time of the failure to prevent it from occurring. The problem is more complex than the one presented in chapter 4, where, in addition to the fact that machines can fail, the individual components belonging to the machines can also fail. Furthermore, in this work we evaluated the use of a multi-agent system, where each agent is responsible for monitoring a single machine, but all agents share the resources, which in this case are the technical resources. Our results showed that the RL policy outperformed the others, reducing the downtime by up to 80% and the downtime prevention by up to 75%. To achieve these results, the RL policy performs more maintenance actions. Finally, we present a first economic evaluation where we show under which parameters one policy outperforms another.

In Chapter 6, we proposed a model of a sustainable maintenance policy that incorporates the three pillars of sustainability. In the economic pillar, we consider the costs of maintenance, breakdowns, and downtime; in the environmental pillar, we consider the carbon footprint given by the operation of the equipment; and in

the social pillar, we consider the fatigue experienced by the technicians. For this purpose, we proposed an evolutionary algorithm based on reinforcement learning, in which several agents learn a maintenance policy with a certain weighting in the sustainability pillars, optimizing with them individually the reward (based on the sustainability pillars), and these agents go through an evolutionary process with the objective of maximizing the hypervolume, thus obtaining a set of agents more evenly distributed among the different objectives. As a result, the policies obtained by our approach show a greater diversity in the objectives of sustainability with respect to classical policies such as CBM and CM that do not adhere to these principles, in addition to showing the trade-off that exists between the economic pillar with respect to the social and environmental pillar, we obtained better results in minimizing the costs.

7.2 Broader implications

To optimize maintenance efficiency, it is essential to understand the condition of the equipment in order to predict potential failures. This requires the use of various technologies, such as sensors that monitor temperature, vibration, pressure, humidity, and other variables in real time. The collection of the data must be transmitted over relliabale networks to ensure both privacy and security. Once collected, the data must be analyzed by algorithms (AI/ML) to identify patterns, correlations, and anomalies that could indicate potential equipment failure. These algorithms must be robust enough to withstand changes in the environment, such as temperature fluctuations, luminosity changes, or vibrations caused by new equipment installation. The use of these technologies, however, represents an additional cost to the industry, including the installation of sensors, network infrastructure, and the computational power required for training and deploying ML models. In addition, incorporating social aspects can lead to enhanced employee well-being and job satisfaction. The integration of technology can also impact workforce-related metrics such as wrench-time which measures the real time technicians spend on maintenance tasks, excluding non-productive activities, by identifying tools, parts or procedure to be done. Furthermore, predictive insights can help identify specific training needs, enabling the development of new skills and improving existing ones, ensuring that technicians are better equipped to operate and understand advanced predictive maintenance systems.

7.3 Limitations and future work

In this thesis, we focus on optimizing maintenance policies in parallel machine environments. We believe that there is still significant scope for further research on the relationship between maintenance and production policies, particularly in heterogeneous environments with production lines and factories. Furthermore, the

policies generated by these agents based on DRL lack transparency, making it difficult to understand the decisions they make. Explainable RL is an important area of focus to ensure that decision-makers have access to more transparent and understandable policies, allowing for more effective adaptation of these models to real industrial environments. We hope that this work will serve as a basis and a source of inspiration for other researchers.

List of publications

Papers included in the dissertation

- M. L. Ruiz Rodríguez et al. Multi-agent deep reinforcement learning based predictive maintenance on parallel machines. *Robotics and Computer-Integrated Manufacturing*, 78:102406, 2022. DOI: 10.1016/j.rcim.2022.102406. URL: https://doi.org/10.1016/j.rcim.2022.102406
- Marcelo Luis Ruiz-Rodríguez et al. Dynamic maintenance scheduling approach under uncertainty: comparison between reinforcement learning, genetic algorithm simheuristic, dispatching rules. *Expert Systems with Applications*, 248:123404, 2024

List of figures

List of tables

Bibliography

[AAA+24] Ahmed M. Attia, Ahmad O. Alatwi, Ahmad Al Hanbali, and Omar G. Alsawafy. Joint maintenance planning and production scheduling optimization model for green environment. *Journal of Quality in Maintenance Engineering*, 30(1):153–174, 2024. ISSN: 13552511. DOI: 10.1108/JQME-05-2023-0047 (cited on pages 4, 22, 24).

[ACZ+20] Youjun An, Xiaohui Chen, Ji Zhang, and Yinghe Li. A hybrid multi-objective evolutionary algorithm to integrate optimization of the production scheduling and imperfect cutting tool maintenance considering total energy consumption. *Journal of Cleaner Production*, 268, September 2020. ISSN: 09596526. DOI: 10.1016/j.jclepro.2020.121540 (cited on pages 21, 24).

[ADI+22] M. Arena, V. Di Pasquale, R. Iannone, S. Miranda, and S. Riemma. A maintenance driven scheduling cockpit for integrated production and maintenance operation schedule. *Advances in Manufacturing*, 10(2):205–219, 2022. DOI: 10.1007/s40436-021-00380-z. URL: https://doi.org/10.1007/s40436-021-00380-z (cited on pages 15, 24).

[AKA24] David Chinalu Anaba, Azeez Jason Kess-Momoh, and Sodrudeen Abolore Ayodeji. Innovative maintenance strategies for industrial equipment: A review of current practices and future directions. *Open Access Research Journal of Science and Technology*, 11(2):029–037, July 2024. ISSN: 2782-9960. DOI: 10.53022/OARJST.2024.11.2.0081. URL: https://oarjst.com/content/innovative-maintenance-strategies-industrial-equipment-review-current-practices-and-future (cited on page 8).

[Al-20] Shireen Al-Hourani. Rescheduling Preventive Maintenance for Utilities Equipment Using Criticality Analysis. *2020 Industrial and Systems Engineering Conference, ISEC 2020*, July 2020. DOI: 10.1109/ISEC49495.2020.9229946 (cited on pages 19, 24).

[AR20] F. F. Alves and M. G. Ravetti. Hybrid proactive approach for solving maintenance and planning problems in the scenario of industry 4.0. *IFAC-PapersOnLine*, 53(3):216–221, 2020. DOI: 10.1016/j.ifacol.2020.11.035. URL: https://doi.org/10.1016/j.ifacol.2020.11.035 (cited on pages 15, 24, 28).

[Arı21] Oğuzhan Ahmet Arık. Population-based tabu search with evolutionary strategies for permutation flow shop scheduling problems under effects of position-dependent learning and linear deterioration. *Soft computing*, 25(2):1501–1518, 2021 (cited on page 74).

[ASjK19] Iqra Asghar, Biswajit Sarkar, and Sung jun Kim. Economic analysis of an integrated production–inventory system under stochastic production capacity and energy consumption. *Energies*, 12(16), August 2019. ISSN: 19961073. DOI: 10.3390/en12163179 (cited on pages 21, 24).

[AX17] Suzan Alaswad and Yisha Xiang. A review on condition-based maintenance optimization models for stochastically deteriorating system. *Reliability engineering & system safety*, 157:54–63, 2017 (cited on page 4).

[BB19] Radhwan Boufellouh and Faycal Belkaid. Multi-objective approach to optimize production and maintenance scheduling in flow shop environment under non-renewable resources constraints. *2019 International Conference on Advanced Electrical Engineering, ICAEE 2019*, November 2019. DOI: 10.1109/ICAEE47123.2019.9014781 (cited on pages 17, 24).

[BDL20] Cristian Bodnar, Ben Day, and Pietro Lió. Proximal distilled evolutionary reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34 of number 04, pages 3283–3290, 2020 (cited on page 79).

[BHK21] Khaoula Ben Abdellafou, Hatem Hadda, and Ouajdi Korbaa. Heuristic algorithms for scheduling intrees on m machines with non-availability constraints. *Operational Research*, 21(1):55–71, March 2021. ISSN: 18661505. DOI: 10.1007/s12351-018-0432-z (cited on pages 17, 24).

[BLA+19] Alexandros Bousdekis, Katerina Lepenioti, Dimitris Apostolou, and Gregoris Mentzas. Decision making in predictive maintenance: Literature review and research agenda for industry 4.0. In *IFAC-PapersOnLine*, volume 52, pages 607–612. Elsevier, January 2019. DOI: 10.1016/j.ifacol.2019.11.226 (cited on page 10).

[BLD22] G. Bencheikh, A. Letouzey, and X. Desforges. An approach for joint scheduling of production and predictive maintenance activities. *Journal of Manufacturing Systems*, 64:546–560, 2022. DOI: 10.1016/j.jmsy.2022.08.005. URL: https://doi.org/10.1016/j.jmsy.2022.08.005 (cited on pages 16, 19, 24).

[BM21] Adil Baykasoğlu and Fatma S. Madenoğlu. Greedy randomized adaptive search procedure for simultaneous scheduling of production and preventive maintenance activities in dynamic flexible job shops. *Soft Computing*, 25(23):14893–14932, December 2021. ISSN: 14337479. DOI: 10.1007/s00500-021-06053-0. URL: https://link-springer-com.proxy.bnl.lu/article/10.1007/s00500-021-06053-0 (cited on pages 17, 24).

[BMM+20] K. Benaggoune, S. Meraghni, J. Ma, L. H. Mouss, and N. Zerhouni. Post prognostic decision for predictive maintenance planning with remaining useful life uncertainty. In *Proceedings - 2020 Prognostics and Health Management Conference, PHM-Besancon 2020*, pages 194–199, 2020. DOI: 10.1109/PHM-Besancon49106.2020.00039. URL: https://doi.org/10.1109/PHM-Besancon49106.2020.00039 (cited on page 28).

[CA21] Meral Calış Duman and Bunyamin Akdemir. A study to determine the effects of industry 4.0 technology components on organizational performance. *Technol Forecast Soc Change*, 167:120615, June 2021. ISSN: 00401625. DOI: 10.1016/j.techfore.2021.120615 (cited on page 8).

[CD20] M. Celen and D. Djurdjanovic. Integrated maintenance and operations decision making with imperfect degradation state observations. *Journal of Manufacturing Systems*, 55:302–316, 2020. DOI: 10.1016/j.jmsy.2020.03.010. URL: https://doi.org/10.1016/j.jmsy.2020.03.010 (cited on pages 15, 24).

[CDC+22] Chris Coleman, Satish Damodaran, Mahesh Chandramouli, and Ed Deuel. Predictive maintenance and the digital supply network, 2022. URL: https://www2.deloitte.com/us/en/insights/focus/industry-4-0/using-predictive-technologies-for-asset-maintenance.html (visited on 04/26/2022) (cited on page 9).

[CGG21] Andrés Cacereño, David Greiner, and Blas J. Galván. Multi-objective optimum design and maintenance of safety systems: An in-depth comparison study including encoding and scheduling aspects with nsga-ii. *Mathematics*, 9(15), August 2021. ISSN: 22277390. DOI: 10.3390/math9151751 (cited on pages 17, 24).

[Cha23] Chin Chih Chang. Optimal preventive replacement last policy for a successive random works system with random lead time. *Communications in Statistics - Theory and Methods*, 52(4):1202–1216, 2023. ISSN: 1532415X. DOI: 10.1080/03610926.2021.1926506. URL: https://www-tandfonline-com.proxy.bnl.lu/doi/abs/10.1080/03610926.2021.1926506 (cited on pages 19, 24).

[CL20] Weiwei Cui and Biao Lu. A bi-objective approach to minimize makespan and energy consumption in flow shops with peak demand constraint. *Sustainability (Switzerland)*, 12(10), May 2020. ISSN: 20711050. DOI: 10.3390/su12104110 (cited on pages 21, 24).

[CL21] Weiwei Cui and Biao Lu. Energy-aware operations management for flow shops under TOU electricity tariff. *Computers and Industrial Engineering*, 151, January 2021. ISSN: 03608352. DOI: 10.1016/j.cie.2020.106942 (cited on pages 21, 24).

[CSX20] Weiwei Cui, Huali Sun, and Beixin Xia. Integrating production scheduling, maintenance planning and energy controlling for the sustainable manufacturing systems under TOU tariff. *Journal of the Operational Research Society*, 71(11):1760–1779, November 2020. ISSN: 14769360. DOI: 10.1080/01605682.2019.1630327. URL: https://www-tandfonline-com.proxy.bnl.lu/doi/abs/10.1080/01605682.2019.1630327 (cited on pages 21, 24).

[CYL+20] Ronghua Chen, Bo Yang, Shi Li, and Shilong Wang. A self-learning genetic algorithm based on reinforcement learning for flexible job-shop scheduling problem. *Computers & industrial engineering*, 149:106778, 2020 (cited on page 79).

[CZS+22] Q. Cao, C. Zanni-Merk, A. Samet, C. Reich, F. B. de Beuvron, A. Beckmann, and C. Giannetti. Kspmi: a knowledge-based system for predictive maintenance in industry 4.0. *Robotics and Computer-Integrated Manufacturing*, 74:102281, 2022 (cited on page 10).

[DB21] Alican Dogan and Derya Birant. Machine learning and data mining in manufacturing. *Expert Systems with Applications*, 166:114060, March 2021. ISSN: 0957-4174. DOI: 10.1016/J.ESWA.2020.114060 (cited on page 50).

[DDC+19] R. Diaz Cazanas, D. R. Delgado Sobrino, D. Caganova, P. Kostal, and K. Velisek. Joint programming of production-maintenance tasks: A simulated annealing-based method. *International Journal of Simulation Modelling*, 18(4):666–677, December 2019. ISSN: 19968566. DOI: 10.2507/IJSIMM18(4)503 (cited on pages 18, 25).

[dGMO+21] Andrea de Giorgio, Antonio Maffei, Mauro Onori, and Lihui Wang. Towards online reinforced learning of assembly sequence planning with interactive guidance systems for industry 4.0 adaptive manufacturing. *Journal of Manufacturing Systems*, 60:22–34, July 2021. ISSN: 0278-6125. DOI: 10.1016/J.JMSY.2021.05.001 (cited on page 50).

[DKT+15] B. De Jonge, W. Klingenberg, R. Teunter, and T. Tinga. Optimum maintenance strategy under uncertainty in the lifetime distribution. *Reliability Engineering and System Safety*, 133:59–67, 2015. DOI: 10.1016/j.ress.2014.09.013. URL: https://doi.org/10.1016/j.ress.2014.09.013 (cited on page 28).

[DNP+19] P. Detti, G. Nicosia, A. Pacifici, and G. Zabalo Manrique de Lara. Robust single machine scheduling with a flexible maintenance activity. *Computers and Operations Research*, 107:19–31, 2019. DOI: 10.1016/j.cor.2019.03.001. URL: https://doi.org/10.1016/j.cor.2019.03.001 (cited on pages 15, 18, 25).

[DR19] Felipe Leno Da Silva and Anna Helena Reali Costa. A survey on transfer learning for multiagent reinforcement learning systems. *Journal of Artificial Intelligence Research*, 64:645–703, 2019. ISSN: 10769757. DOI: 10.1613/jair.1.11396 (cited on pages 52, 62).

[Dru19] Madalina M. Drugan. Reinforcement learning versus evolutionary computation: A survey on hybrid algorithms. *Swarm and Evolutionary Computation*, 44:228–246, February 2019. ISSN: 2210-6502. DOI: 10.1016/J.SWEVO.2018.03.011 (cited on pages 62, 79).

[DS19] Manocher Djassemi and Hamid Seifoddini. Analysis of critical machine reliability in manufacturing cells. *Journal of Industrial Engineering and Management*, 12(1):70–82, 2019. ISSN: 20130953. DOI: 10.3926/jiem.2757 (cited on pages 19, 25).

[DS20] B. De Jonge and P. A. Scarf. A review on maintenance optimization. *European Journal of Operational Research*, 285(3):805–824, 2020. DOI: 10.1016/j.ejor.2019.09.047. URL: https://doi.org/10.1016/j.ejor.2019.09.047 (cited on pages 3, 28, 29).

[DS98] I. T. Dedopoulos and Y. Smeers. An age reduction approach for finite horizon optimization of preventive maintenance for single units subject to random failures. *Computers and Industrial Engineering*, 34(3):643–654, 1998. ISSN: 03608352. DOI: 10.1016/S0360-8352(97)00281-7 (cited on page 58).

[dWGM+20] Christian Schroeder de Witt, Tarun Gupta, Denys Makoviichuk, Viktor Makoviychuk, Philip HS Torr, Mingfei Sun, and Shimon Whiteson. Is independent learning all you need in the starcraft multi-agent challenge? *arXiv preprint arXiv:2011.09533*, 2020 (cited on page 62).

[DY20] J. Dong and C. Ye. Research on two-stage joint optimization problem of green manufacturing and maintenance for semiconductor wafer. *Mathematical Problems in Engineering*, 2020, 2020 (cited on pages 22, 25).

[DZY+24] S. Deng, Y. Zhu, Y. Yu, and X. Huang. An integrated approach of ensemble learning methods for stock index prediction using investor sentiments. *Expert Systems with Applications*, 238:121710, 2024. DOI: 10.1016/j.eswa.2023.121710. URL: https://doi.org/10.1016/j.eswa.2023.121710 (cited on page 28).

[Ele24] Electricity Maps. Data Portal, 2024. URL: https://www.electricitymaps.com/data-portal. Accessed: 2024-07-17 (cited on page 79).

[Eur19] European Commission. The European Green Deal. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A52019DC0640, 2019. Communication from the Commission to the European Parliament, the Council, the European Economic and Social Committee and the Committee of the Regions. Accessed on 16 July 2024 (cited on page 68).

[FAP+17] A. Ferjani, A. Ammar, H. Pierreval, and S. Elkosantini. A simulation-optimization based heuristic for the online assignment of multi-skilled workers subjected to fatigue in manufacturing systems. *Computers and Industrial Engineering*, 112:663–674, 2017. DOI: 10.1016/j.cie.2017.02.008. URL: https://doi.org/10.1016/j.cie.2017.02.008 (cited on pages 3, 28, 30).

[FFA+18] Jakob N Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual Multi-Agent Policy Gradients. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1), April 2018. ISSN: 2374-3468. URL: https://ojs.aaai.org/index.php/AAAI/article/view/11794 (cited on pages 61, 62).

[FGC+22] Tianfan Fu, Wenhao Gao, Connor Coley, and Jimeng Sun. Reinforced genetic algorithm for structure-based drug design. *Advances in Neural Information Processing Systems*, 35:12325–12338, 2022 (cited on page 79).

[FTS19] A. Farahani, H. Tohidi, and A. Shoja. An integrated optimization of quality control chart parameters and preventive maintenance using markov chain. *Advances in production engineering & management*, 14(1), 2019 (cited on pages 18, 25).

[GDA+23] M. Geurtsen, J. B. H. C. Didden, J. Adan, Z. Atan, and I. Adan. Production, maintenance and resource scheduling: a review. *European Journal of Operational Research*, 305(2):501–529, 2023. DOI: 10.1016/j.ejor.2022.03.045. URL: https://doi.org/10.1016/j.ejor.2022.03.045 (cited on pages 15, 28).

[GJ21] Shrajal Gupta and Ajai Jain. Assessing the Effect of Reliability-Based Maintenance Approach in Job Shop Scheduling with Setup Time and Energy Consideration Using Simulation; A Simulation Study. *Smart Science*, 9(4):283–304, 2021. ISSN: 23080477. DOI: 10.1080/23080477.2021.1938502. URL: https://www-tandfonline-com.proxy.bnl.lu/doi/abs/10.1080/23080477.2021.1938502 (cited on pages 22, 25).

[GJ22] Shrajal Gupta and Ajai Jain. Analysis of Integrated Preventive Maintenance and Machine Failure in Stochastic Flexible Job Shop Scheduling with Sequence-dependent Setup Time. *Smart Science*, 10(3):175–197, 2022. ISSN: 23080477. DOI: 10.1080/23080477.2021.1992823. URL: https://www-tandfonline-com.proxy.bnl.lu/doi/abs/10.1080/23080477.2021.1992823 (cited on pages 20, 25).

[GJC+23] Shrajal Gupta, Ajai Jain, Felix T.S. Chan, and Rakesh Kumar Phanden. A study on simulation-based optimization of a stochastic flexible job shop scheduling undergoing preventive maintenance with sequence-dependent setup time. *International Journal on Interactive Design and Manufacturing*, 2023. ISSN: 19552505. DOI: 10.1007/s12008-023-01651-8. URL: https://doi.org/10.1007/s12008-023-01651-8 (cited on pages 20, 25).

[GLG+21] Jakob Giner, Raphael Lamprecht, Viola Gallina, Catherine Laflamme, Lennard Sielaff, and Wilfried Sihn. Demonstrating Reinforcement Learning for Maintenance Scheduling in a Production Environment. *IEEE International Conference on Emerging Technologies and Factory Automation, ETFA*, 2021-Septe, 2021. ISSN: 19460759. DOI: 10.1109/ETFA45728.2021.9613205 (cited on pages 18, 25).

[GP21] Christopher AK Gordon and Efstratios N Pistikopoulos. Data-driven prescriptive maintenance toward fault-tolerant multiparametric control. *AIChE Journal*:e17489, 2021 (cited on page 10).

[GS18] Maheshwaran Gopalakrishnan and Anders Skoogh. Machine criticality based maintenance prioritization: identifying productivity improvement potential. *International Journal of Productivity and Performance Management*, 67(4):654–672, 2018 (cited on page 4).

[GTS+20] M. Ghaleb, S. Taghipour, M. Sharifi, and H. Zolfagharinia. Integrated production and maintenance scheduling for a single degrading machine with deterioration-based failures. *Computers and Industrial Engineering*, 143:106432, 2020. DOI: 10.1016/j.cie.2020.106432. URL: https://doi.org/10.1016/j.cie.2020.106432 (cited on pages 21, 25).

[GZT+23] S. Guan, Z. Zhuang, H. Tao, Y. Chen, V. Stojanovic, and W. Paszke. 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, 2023. DOI: 10.1177/01423312221142564. URL: https://doi.org/10.1177/01423312221142564 (cited on page 28).

[HAG+21] Lotfi Hidri, Khaled Alqahtani, Achraf Gazdar, and Ahmed Badwelan. Integrated Scheduling of Tasks and Preventive Maintenance Periods in a Parallel Machine Environment with Single Robot Server. *IEEE Access*, 9:74454–74470, 2021. ISSN: 21693536. DOI: 10.1109/ACCESS.2021.3081495 (cited on pages 17, 25).

[Hal93] Arthur J. Hallinan. A Review of the Weibull Distribution. *Journal of Quality Technology*, 25(2):85–93, 1993. ISSN: 0022-4065. DOI: 10.1080/00224065.1993.11979431 (cited on page 54).

[HCA20] Jing Huang, Qing Chang, and Jorge Arinez. Deep reinforcement learning based preventive maintenance policy for serial production lines. *Expert Systems with Applications*, 160:113701, December 2020. ISSN: 09574174. DOI: 10.1016/j.eswa.2020.113701 (cited on pages 30, 52).

[HCC19] Jing Huang, Qing Chang, and Nilanjan Chakraborty. Machine preventive replacement policy for serial production lines based on reinforcement learning. In *IEEE International Conference on Automation Science and Engineering*, volume 2019-Augus, pages 523–528. IEEE, 2019. ISBN: 9781728103556. DOI: 10.1109/COASE.2019.8843338 (cited on page 52).

[HFR21] Seyed Mohammad Hadian, Hiwa Farughi, and Hasan Rasay. Joint planning of maintenance, buffer stock and quality control for unreliable, imperfect manufacturing systems. *Computers & Industrial Engineering*, 157:107304, 2021 (cited on page 62).

[HLB19] Djalal Hedjazi, Fateh Layachi, and Djallel Eddine Boubiche. A multi-agent system for distributed maintenance scheduling. *Computers and Electrical Engineering*, 77:1–11, July 2019. ISSN: 00457906. DOI: 10.1016/j.compeleceng.2019.04.016 (cited on pages 18, 25).

[JAA21] R. P.S.K. Jayasuriya, P. A.G.M. Amarasinghe, and S. K. Abeygunawardane. Application of artificial intelligence for maintenance modelling of critical machines in solid tire manufacturing. *2021 International Conference on Innovative Trends in Information Technology, ICITIIT 2021*, February 2021. DOI: 10.1109/ICITIIT51526.2021.9399600 (cited on pages 18, 25).

[JDR+06] IS Jawahir, OW Dillon, KE Rouch, Kunal J Joshi, Anand Venkatachalam, and Israd H Jaafar. Total life-cycle considerations in product design for sustainability: a framework for comprehensive evaluation. In *Proceedings of the 10th international research/expert conference, Barcelona, Spain*, volume 1 of number 10. Citeseer, 2006 (cited on page 71).

[JGN13] Mohamad Y Jaber, ZS Givi, and W Patrick Neumann. Incorporating human fatigue and recovery into the learning–forgetting process. *Applied Mathematical Modelling*, 37(12-13):7287–7299, 2013 (cited on page 75).

[JKF+19] Zoe Juozapaitis, Anurag Koul, Alan Fern, Martin Erwig, and Finale Doshi-Velez. Explainable Reinforcement Learning via Reward Decomposition. *Proceedings of the IJCAI 2019 Workshop on Explainable Artificial Intelligence*:47–53, 2019 (cited on page 62).

[JLK20] Małgorzata Jasiulewicz-Kaczmarek, Stanislaw Legutko, and Piotr Kluk. Maintenance 4.0 technologies–new opportunities for sustainability driven maintenance. *Management and production engineering review*, 11, 2020 (cited on page 4).

[JM21] S. Jayatilleka and J. McLinn. Practical implications of weibull shape parameter; lessons & pitfalls. In *2021 Annual Reliability and Maintainability Symposium (RAMS)*, pages 1–6, 2021. DOI: 10.1109/RAMS48097.2021.9605732. URL: https://doi.org/10.1109/RAMS48097.2021.9605732 (cited on page 37).

[KBL+23] E. Kaufmann, L. Bauersfeld, A. Loquercio, M. Müller, V. Koltun, and D. Scaramuzza. Champion-level drone racing using deep reinforcement learning. *Nature*, 620(7976):982–987, 2023. DOI: 10.1038/s41586-023-06419-4. URL: https://doi.org/10.1038/s41586-023-06419-4 (cited on page 28).

[KJL19] A. Kuhnle, J. Jakubik, and G. Lanza. Reinforcement learning for opportunistic maintenance optimization. *Production Engineering*, 13(1):33–41, 2019. DOI: 10.1007/s11740-018-0855-7. URL: https://doi.org/10.1007/s11740-018-0855-7 (cited on pages 16, 25, 52, 56, 62).

[KLS+20] Yun Geon Kim, Seokgi Lee, Jiyeon Son, Heechul Bae, and Byung Do Chung. Multi-agent system and reinforcement learning approach for distributed intelligence in a flexible smart manufacturing system. *Journal of Manufacturing Systems*, 57:440–450, 2020 (cited on page 50).

[KLV+22] Eda Köksal Ahmed, Zengxiang Li, Bharadwaj Veeravalli, and Shen Ren. Reinforcement learning-enabled genetic algorithm for school bus scheduling. *Journal of Intelligent Transportation Systems*, 26(3):269–283, 2022 (cited on page 79).

[Kom02] Kari Komonen. A cost model of industrial maintenance for profitability analysis and benchmarking. *Int J Prod Econ*, 79(1):15–31, September 2002. ISSN: 09255273. DOI: 10.1016/S0925-5273(00)00187-0 (cited on page 8).

[KPN+24] Ghislain Serge Mepouly Kedy, Malcom Chumchoua Penda, Léandre Nneme Nneme, Olivier Thierry Sosso Mayi, and Léopold Gustave Lehman. Enhancing the effectiveness of joint production and maintenance scheduling based on a multi-agent system and a Pigouvian approach of externalities. *Production Engineering*, (0123456789), 2024. ISSN: 18637353. DOI: 10.1007/s11740-024-01272-4. URL: https://doi.org/10.1007/s11740-024-01272-4 (cited on pages 19, 25).

[LCZ21] B. Lu, Z. Chen, and X. Zhao. Data-driven dynamic predictive maintenance for a manufacturing system with quality deterioration and online sensors. *Reliability Engineering and System Safety*, 212:107628, 2021. DOI: 10.1016/j.ress.2021.107628. URL: https://doi.org/10.1016/j.ress.2021.107628 (cited on pages 15, 25).

[LDC18] Q. Liu, M. Dong, and F. F. Chen. Single-machine-based joint optimization of predictive maintenance planning and production scheduling. *Robotics and Computer-Integrated Manufacturing*, 51:238–247, 2018 (cited on page 10).

[Lee96] Chung-Yee Lee. Machine scheduling with an availability constraint. *Journal of global optimization*, 9(3):395–416, 1996 (cited on page 68).

[LHY+20] W Luo, T Hu, Y Ye, C Zhang, and Y Wei. A hybrid predictive maintenance approach for CNC machine tool driven by Digital Twin. *Robot Comput Integr Manuf*, 65:101974, 2020 (cited on page 9).

[LPB+20] Katerina Lepenioti, Minas Pertselakis, Alexandros Bousdekis, Andreas Louca, Fenareti Lampathaki, Dimitris Apostolou, Gregoris Mentzas, and Stathis Anastasiou. Machine Learning for Predictive and Prescriptive Analytics of Operational Data in Smart Manufacturing. *Lecture Notes in Business Information Processing*, 382 LNBIP:5–16, June 2020. ISSN: 18651356. DOI: 10.1007/978-3-030-49165-9_1 (cited on page 50).

[LRC+23] Xinlong Li, Yan Ran, Baojia Chen, Fafa Chen, Yunfei Cai, and Genbao Zhang. Opportunistic maintenance strategy optimization considering imperfect maintenance under hybrid unit-level maintenance strategy. *Computers and Industrial Engineering*, 185(September):109624, 2023. ISSN: 03608352. DOI: 10.1016/j.cie.2023.109624. URL: https://doi.org/10.1016/j.cie.2023.109624 (cited on pages 20, 25).

[LWN+23] I. D. Lutz, S. Wang, C. Norn, A. Courbet, A. J. Borst, Y. T. Zhao, A. Dosey, L. Cao, J. Xu, E. M. Leaf, C. Treichel, P. Litvicov, P. Li, A. D. Goodson, P. Rivera-Sánchez, A. M. Bratovianu, M. Baek, N. P. King, H. Ruohola-Baker, and D. Baker. Top-down design of protein architectures with reinforcement learning. *Science*, 380(6642):266–273, 2023. DOI: 10.1126/science.adf6591. URL: https://doi.org/10.1126/science.adf6591 (cited on page 28).

[MDZ+22] B. Miao, Q. Deng, L. Zhang, Z. Huo, and X. Liu. Collaborative scheduling of spare parts production and service workers driven by distributed maintenance demand. *Journal of Manufacturing Systems*, 64:261–274, 2022. DOI: 10.1016/j.jmsy.2022.06.012. URL: https://doi.org/10.1016/j.jmsy.2022.06.012 (cited on pages 14, 25).

[MFZ+20] S. Mi, Y. Feng, H. Zheng, Z. Li, Y. Gao, and J. Tan. Integrated intelligent green scheduling of predictive maintenance for complex equipment based on information services. *IEEE Access*, 8:45797–45812, 2020. DOI: 10.1109/ACCESS.2020.2977667. URL: https://doi.org/10.1109/ACCESS.2020.2977667 (cited on pages 22, 25).

[MGC+22] D. Mukherjee, K. Gupta, L. H. Chang, and H. Najjaran. A survey of robot learning strategies for human-robot collaboration in industrial settings. *Robotics and Computer-Integrated Manufacturing*, 73:102231, 2022 (cited on page 62).

[MKS+13] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013 (cited on pages 11, 79).

[MMS+20] Prashan Madumal, Tim Miller, Liz Sonenberg, and Frank Vetere. Explainable reinforcement learning through a causal lens. *AAAI 2020 - 34th AAAI Conference on Artificial Intelligence*:2493–2500, 2020. ISSN: 2159-5399. DOI: 10.1609/aaai.v34i03.5631. arXiv: 1905.10958 (cited on page 62).

[Mni16] Volodymyr Mnih. Asynchronous methods for deep reinforcement learning. *arXiv preprint arXiv:1602.01783*, 2016 (cited on page 12).

[Moh18] Mamad Mohamed. Challenges and benefits of industry 4.0: an overview. *International Journal of Supply and Operations Management*, 5(3):256–265, 2018 (cited on page 2).

[Mor20] Miguel Morales. *Grokking deep reinforcement learning*. Manning Publications, 2020 (cited on page 11).

[MT19] Nasim Mirahmadi and Sharareh Taghipour. Energy-efficient optimization of flexible job shop scheduling and preventive maintenance. *Proceedings - Annual Reliability and Maintainability Symposium*, 2019-Janua, 2019. ISSN: 0149144X. DOI: 10.1109/RAMS.2019.8768908 (cited on pages 21, 25).

[NGY+23] A. Neumann, S. Gounder, X. Yan, G. Sherman, B. Campbell, M. Guo, and F. Neumann. Diversity optimization for the detection and concealment of spatially defined communication networks. In *GECCO 2023 - Proceedings of the 2023 Genetic and Evolutionary Computation Conference*, pages 1436–1444, 2023. DOI: 10.1145/3583131.3590405. URL: https://doi.org/10.1145/3583131.3590405 (cited on page 28).

[NKT+20] I. Németh, Á. Kocsis, D. Takács, B. W. Shaheen, M. Takács, A. Merlo, A. Eytan, L. Bidoggia, and P. Olocco. Maintenance schedule optimisation for manufacturing systems. *IFAC-PapersOnLine*, 53(3):319–324, 2020. DOI: 10.1016/j.ifacol.2020.11.051. URL: https://doi.org/10.1016/j.ifacol.2020.11.051 (cited on pages 14, 25, 35).

[NNA+18] Nasruddin, S. Nasution, N. Aisyah, A. Surachman, and A. S. Wibowo. Exergy analysis and exergoeconomic optimization of a binary cycle system using a multi objective genetic algorithm. *International Journal of Technology*, 9(2):275–286, 2018. DOI: 10.14716/ijtech.

v9i2.1040. URL: https://doi.org/10.14716/ijtech.v9i2.1040 (cited on page 28).

[NNN20] Thanh Thi Nguyen, Ngoc Duy Nguyen, and Saeid Nahavandi. Deep Reinforcement Learning for Multiagent Systems: A Review of Challenges, Solutions, and Applications. *IEEE Transactions on Cybernetics*, 50(9):3826–3839, 2020. ISSN: 21682275. DOI: 10.1109/TCYB.2020.2977374. arXiv: 1812.11794 (cited on pages 52, 62).

[PAE+17] Deepak Pathak, Pulkit Agrawal, Alexei A. Efros, and Trevor Darrell. Curiosity-driven Exploration by Self-supervised Prediction, July 2017 (cited on page 59).

[Pal19] Richard D Palmer. *Maintenance planning and scheduling handbook*. McGraw-Hill New York, 2019 (cited on page 62).

[PKK20] Panagiotis D. Paraschos, Georgios K. Koulinas, and Dimitrios E. Koulouriotis. Reinforcement learning for combined production-maintenance and quality control of a manufacturing system with deterioration failures. *Journal of Manufacturing Systems*, 56:470–483, July 2020. ISSN: 02786125. DOI: 10.1016/j.jmsy.2020.07.004 (cited on pages 50, 62).

[PKK23] Panagiotis D. Paraschos, Georgios K. Koulinas, and Dimitrios E. Koulouriotis. A reinforcement learning/ad-hoc planning and scheduling mechanism for flexible and sustainable manufacturing systems. *Flexible Services and Manufacturing Journal*, 2023. ISSN: 19366590. DOI: 10.1007/s10696-023-09496-9 (cited on pages 22, 25).

[PL23] W. Pan and S. Q. Liu. Deep reinforcement learning for the dynamic and uncertain vehicle routing problem. *Applied Intelligence*, 53(1):405–422, 2023. DOI: 10.1007/s10489-022-03456-w. URL: https://doi.org/10.1007/s10489-022-03456-w (cited on page 28).

[PMA23] Foivos Psarommatis, Gökan May, and Victor Azamfirei. Envisioning maintenance 5.0: insights from a systematic literature review of industry 4.0 and a proposed framework. *Journal of Manufacturing Systems*, 68:376–399, 2023 (cited on page 69).

[QAA+20] Ammar A. Qamhan, Aref Ahmed, Ibrahim M. Al-Harkan, Ahmed Badwelan, Ali M. Al-Samhan, and Lotfi Hidri. An Exact Method and Ant Colony Optimization for Single Machine Scheduling Problem with Time Window Periodic Maintenance. *IEEE Access*, 8:44836–44845, 2020. ISSN: 21693536. DOI: 10.1109/ACCESS.2020.2977234 (cited on pages 17, 25).

[QZL+22] W. Qin, Z. Zhuang, Y. Liu, and J. Xu. Sustainable service oriented equipment maintenance management of steel enterprises using a two-stage optimization approach. *Robotics and Computer-Integrated Manufacturing*, 75:102311, 2022 (cited on pages 22, 25).

[RF21] Pegah Rokhforoz and Olga Fink. 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, 2021 (cited on pages 15, 25).

[RF22] Pegah Rokhforoz and Olga Fink. Maintenance scheduling of manufacturing systems based on optimal price of the network. *Reliability Engineering and System Safety*, 217, January 2022. ISSN: 09518320. DOI: 10.1016/j.ress.2021.108088 (cited on pages 19, 25).

[RKdG+22] M. L. Ruiz Rodríguez, S. Kubler, A. de Giorgio, M. Cordy, J. Robert, and Y. Le Traon. Multi-agent deep reinforcement learning based predictive maintenance on parallel machines. *Robotics and Computer-Integrated Manufacturing*, 78:102406, 2022. DOI: 10.1016/j.rcim.2022.102406. URL: https://doi.org/10.1016/j.rcim.2022.102406 (cited on pages 30, iii).

[RKR+24] Marcelo Luis Ruiz-Rodríguez, Sylvain Kubler, Jérémy Robert, and Yves Le Traon. Dynamic maintenance scheduling approach under uncertainty: comparison between reinforcement learning, genetic algorithm simheuristic, dispatching rules. *Expert Systems with Applications*, 248:123404, 2024 (cited on pages 68, iii).

[RSD+18] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder De Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. QMIX: Monotonic value function factorisation for deep multi-agent reinforcement Learning. In *35th International Conference on Machine Learning, ICML 2018*, volume 10, pages 6846–6859, 2018. ISBN: 9781510867963. arXiv: 2003.08839 (cited on page 62).

[RSL20] E. Ruschel, E. A. P. Santos, and E. de F. R. Loures. Establishment of maintenance inspection intervals: an application of process mining techniques in manufacturing. *Journal of Intelligent Manufacturing*, 31(1):53–72, 2020. DOI: 10.1007/s10845-018-1434-7. URL: https://doi.org/10.1007/s10845-018-1434-7 (cited on pages 15, 25).

[RZL+19] Yongyi Ran, Xin Zhou, Pengfeng Lin, Yonggang Wen, and Ruilong Deng. A survey of predictive maintenance: systems, purposes and approaches. *arXiv preprint arXiv:1912.07383*, 2019 (cited on page 68).

[SAS21] Sebastian Schwendemann, Zubair Amjad, and Axel Sikora. A survey of machine-learning techniques for condition monitoring and predictive maintenance of bearings in grinding machines. *Computers in Industry*, 125:103380, February 2021. ISSN: 0166-3615. DOI: 10.1016/J.COMPIND.2020.103380 (cited on page 50).

[SB18] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018 (cited on pages 10, 11, 50, 54, 76).

[SBS+21] A. Sarazin, J. Bascans, J. B. Sciau, J. Song, B. Supiot, A. Montarnal, X. Lorca, and S. Truptil. Expert system dedicated to condition-based maintenance based on a knowledge graph approach: application to an aeronautic system. *Expert Systems with Applications*, 186:115767, 2021. DOI: 10.1016/j.eswa.2021.115767 (cited on pages 3, 30).

[SC20] In Ho Sin and Byung Do Chung. Bi-objective optimization approach for energy aware scheduling considering electricity cost and preventive maintenance using genetic algorithm. *Journal of Cleaner Production*, 244, January 2020. ISSN: 09596526. DOI: 10.1016/j.jclepro.2019.118869 (cited on pages 21, 25).

[Sch15] John Schulman. Trust region policy optimization. *arXiv preprint arXiv:1502.05477*, 2015 (cited on page 11).

[SCO+20] CY Siew, Miko May Lee Chang, Soh-Khim Ong, and Andrew YC Nee. Human-oriented maintenance and disassembly in sustainable manufacturing. *Computers & Industrial Engineering*, 150:106903, 2020 (cited on page 4).

[SDL20] Zeyi Sun, Fadwa Dababneh, and Lin Li. Joint Energy, Maintenance, and Throughput Modeling for Sustainable Manufacturing Systems. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 50(6):2101–2112, 2020. ISSN: 21682232. DOI: 10.1109/TSMC.2018.2799740 (cited on pages 4, 21, 25).

[SFZ20] Hany Seidgar, Hamed Fazlollahtabar, and Mostafa Zandieh. Scheduling two-stage assembly flow shop with random machines breakdowns: integrated new self-adapted differential evolutionary and simulation approach. *Soft Computing*, 24(11):8377–8401, June 2020. ISSN: 14337479. DOI: 10.1007/s00500-019-04407-3. URL: https://link-springer-com.proxy.bnl.lu/article/10.1007/s00500-019-04407-3 (cited on pages 16, 25).

[SHA+22] Jianyu Su, Jing Huang, Stephen Adams, Qing Chang, and Peter A Beling. Deep multi-agent reinforcement learning for multi-level preventive maintenance in manufacturing systems. *Expert systems with applications*, 192:116323, 2022 (cited on pages 28, 52).

[SML+21] Anurag Shrivastava, K. Murali Krishna, Moti Lal Rinawa, Mukesh Soni, Gowtham Ramkumar, and Sushma Jaiswal. Inclusion of IoT, ML, and Blockchain Technologies in Next Generation Industry 4.0 Environment. *Mater Today Proc*, July 2021. ISSN: 22147853. DOI: 10.1016/j.matpr.2021.07.273 (cited on page 8).

[SPC+21] Ho Chit Siu, Jaime Peña, Edenna Chen, Yutai Zhou, Victor Lopez, Kyle Palko, Kimberlee Chang, and Ross Allen. Evaluation of human-ai teams for learned and rule-based agents in hanabi. *Advances in Neural Information Processing Systems*, 34, 2021 (cited on page 62).

[SS22] P. Sulewski and M. Szymkowiak. The weibull lifetime model with randomised failure-free time. *Statistics in Transition New Series*, 23(4):59–76, 2022. DOI: 10.2478/stattrans-2022-0042. URL: https://doi.org/10.2478/stattrans-2022-0042 (cited on page 36).

[SSS+23] X. Song, P. Sun, S. Song, and V. Stojanovic. Finite-time adaptive neural resilient dsc for fractional-order nonlinear large-scale systems against sensor-actuator faults. *Nonlinear Dynamics*, 111(13):12181–12196, 2023. DOI: 10.1007/s11071-023-08456-0. URL: https://doi.org/10.1007/s11071-023-08456-0 (cited on page 28).

[ST21] Mani Sharifi and Sharareh Taghipour. Optimal production and maintenance scheduling for a degrading multi-failure modes single-machine production environment. *Applied Soft Computing*, 106:107312, July 2021. ISSN: 15684946. DOI: 10.1016/j.asoc.2021.107312 (cited on pages 21, 25).

[Sto23] V. Stojanovic. Fault-tolerant control of a hydraulic servo actuator via adaptive dynamic programming. *Mathematical Modelling and Control*, 3(3):181–191, 2023. DOI: 10.3934/mmc.2023016. URL: https://doi.org/10.3934/mmc.2023016 (cited on page 28).

[SWD+17] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal Policy Optimization Algorithms, July 2017. arXiv: 1707.06347 (cited on pages 12, 34, 41, 59).

[SWY+23] Yanjie Song, Luona Wei, Qing Yang, Jian Wu, Lining Xing, and Yingwu Chen. Rl-ga: a reinforcement learning-based genetic algorithm for electromagnetic detection satellite scheduling problem. *Swarm and Evolutionary Computation*, 77:101236, 2023 (cited on page 79).

[SYP19] Q. Sun, Z. S. Ye, and W. Peng. Scheduling preventive maintenance considering the saturation effect. *IEEE Transactions on Reliability*, 68(2):741–752, 2019. DOI: 10.1109/TR.2018.2874265. URL: https://doi.org/10.1109/TR.2018.2874265 (cited on pages 14, 25).

[SZ19] Jiayu Shen and Yuanguo Zhu. A parallel-machine scheduling problem with periodic maintenance under uncertainty. *Journal of Ambient Intelligence and Humanized Computing*, 10(8):3171–3179, August 2019. ISSN: 18685145. DOI: 10.1007/s12652-018-1032-8 (cited on pages 20, 25).

[TDF+21] Alessia MR Tortora, Valentina Di Pasquale, Chiara Franciosi, Salvatore Miranda, and Raffaele Iannone. The role of maintenance operator in industrial manufacturing systems: research topics and trends. *Applied Sciences*, 11(7):3193, 2021 (cited on page 4).

[TQL+18] Fei Tao, Qinglin Qi, Ang Liu, and Andrew Kusiak. Data-driven smart manufacturing. *Journal of Manufacturing Systems*, 48:157–169, July 2018. ISSN: 0278-6125. DOI: 10.1016/J.JMSY.2018.01.006 (cited on page 9).

[Uni15] United Nations. Transforming our world: the 2030 Agenda for Sustainable Development, 2015. URL: https://undocs.org/en/A/RES/70/1. General Assembly document A/RES/70/1 (cited on page 68).

[VAW+22] A. Valet, T. Altenmüller, B. Waschneck, M. C. May, A. Kuhnle, and G. Lanza. Opportunistic maintenance scheduling with deep reinforcement learning. *Journal of Manufacturing Systems*, 64:518–534, 2022. DOI: 10.1016/j.jmsy.2022.07.016. URL: https://doi.org/10.1016/j.jmsy.2022.07.016 (cited on pages 3, 16, 25).

[WBF+20] M. Wocker, N. K. Betz, C. Feuersänger, A. Lindworsky, and J. Deuse. Unsupervised learning for opportunistic maintenance optimization in flexible manufacturing systems. *Procedia CIRP*, 93:1025–1030, 2020. DOI: 10.1016/j.procir.2020.04.025. URL: https://doi.org/10.1016/j.procir.2020.04.025 (cited on pages 14, 25, 30).

[WDX+20] Junkai Wang, Hangming Du, Junxia Xing, Fei Qiao, and Yumin Ma. Novel Energy- And Maintenance-Aware Collaborative Scheduling for A Hybrid Flow Shop Based on Dual Memetic Algorithms. *IEEE Robotics and Automation Letters*, 5(4):5613–5620, October 2020. ISSN: 23773766. DOI: 10.1109/LRA.2020.3005626 (cited on pages 21, 25).

[WDZ+23] Zhen Wang, Qianwang Deng, Like Zhang, Haiqiu Li, and Fengyuan Li. Joint optimization of integrated mixed maintenance and distributed two-stage hybrid flow-shop production for multi-site maintenance requirements. *Expert Systems with Applications*, 215, April 2023. ISSN: 09574174. DOI: 10.1016/j.eswa.2022.119422 (cited on pages 17, 25).

[WSU22] Harjanti Widiastuti, Adelia Sulistyani, and Evy Rahman Utami. Do environmental issues matter to investors? In *International Conference on Sustainable Innovation Track Accounting and Management Sciences (ICOSIAMS 2021)*, pages 228–234. Atlantis Press, 2022 (cited on page 71).

[WXZ+22] Junliang Wang, Chuqiao Xu, Jie Zhang, and Ray Zhong. Big data analytics for intelligent manufacturing systems: a review. *Journal of Manufacturing Systems*, 62:738–752, 2022 (cited on page 68).

[WYD+20] Cheng Hung Wu, Yi Chun Yao, Stéphane Dauzère-Pérès, and Cheng Juei Yu. Dynamic dispatching and preventive maintenance for parallel machines with dispatching-dependent deterioration. *Computers and Operations Research*, 113:104779, January 2020. ISSN: 03050548. DOI: 10.1016/j.cor.2019.104779 (cited on pages 20, 25).

[WYW21] Hongfeng Wang, Qi Yan, and Junwei Wang. Blockchain-secured multi-factory production with collaborative maintenance using Q learning-based optimisation approach. *International Journal of Production Research*, 2021. ISSN: 1366588X. DOI: 10.1080/00207543.2021.2002968 (cited on page 51).

[XC19] W. Xu and L. Cao. Optimal maintenance control of machine tools for energy efficient manufacturing. *The International Journal of Advanced Manufacturing Technology*, 104(9):3303–3311, 2019 (cited on page 50).

[XKK+17] A. S. Xanthopoulos, Athanasios Kiatipis, D. E. Koulouriotis, and Sepp Stieger. Reinforcement Learning-Based and Parametric Production-Maintenance Control Policies for a Deteriorating Manufacturing System. *IEEE Access*, 6:576–588, November 2017. ISSN: 21693536. DOI: 10.1109/ACCESS.2017.2771827 (cited on page 50).

[XSS+21] T. Xia, G. Shi, G. Si, S. Du, and L. Xi. Energy-oriented joint optimization of machine maintenance and tool replacement in sustainable manufacturing. *Journal of Manufacturing Systems*, 59:261–271, 2021. DOI: 10.1016/j.jmsy.2021.01.015. URL: https://doi.org/10.1016/j.jmsy.2021.01.015 (cited on pages 4, 22, 25).

[XSS+22] Tangbin Xia, Guojin Si, Guo Shi, Kaigan Zhang, and Lifeng Xi. Optimal selective maintenance scheduling for series–parallel systems based on energy efficiency optimization. *Applied Energy*, 314:118927, May 2022. ISSN: 03062619. DOI: 10.1016/j.apenergy.2022.118927 (cited on pages 22, 25).

[YAK+22] R. Yazdani, M. Alipour-Vaezi, K. Kabirifar, A. Salahi Kojour, and F. Soleimani. 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, 2022. DOI: 10.1007/s00500-022-07436-7. URL: https://doi.org/10.1007/s00500-022-07436-7 (cited on pages 14, 25).

[YH21] Tae Sun Yu and Jun Hee Han. Scheduling proportionate flow shops with preventive machine maintenance. *International Journal of Production Economics*, 231:107874, January 2021. ISSN: 09255273. DOI: 10.1016/j.ijpe.2020.107874 (cited on pages 20, 25).

[YLC16] K. C. Ying, C. C. Lu, and J. C. Chen. Exact algorithms for single-machine scheduling problems with a variable maintenance. *Computers and Industrial Engineering*, 98:427–433, 2016. DOI: 10.1016/j.cie.2016.05.037. URL: https://doi.org/10.1016/j.cie.2016.05.037 (cited on page 28).

[YRZ+22] X. Yang, Y. Ran, G. Zhang, H. Wang, Z. Mu, and S. Zhi. A digital twin-driven hybrid approach for the prediction of performance degradation in transmission unit of cnc machine tool. *Robotics and Computer-Integrated Manufacturing*, 73:102230, 2022 (cited on page 10).

[YSW+23] C. Y. Yang, C. Shiranthika, C. Y. Wang, K. W. Chen, and S. Sumathipala. Reinforcement learning strategies in cancer chemotherapy treatments: a review. *Computer Methods and Programs in Biomedicine*, 229:107280, 2023. DOI: 10.1016/j.cmpb.2022.107280. URL: https://doi.org/10.1016/j.cmpb.2022.107280 (cited on page 28).

[YTC22] Nooshin Yousefi, Stamatis Tsianikas, and David W Coit. Dynamic maintenance model for a repairable multi-component system using deep reinforcement learning. *Quality Engineering*, 34(1):16–35, 2022 (cited on page 4).

[YWW22] Q. Yan, H. Wang, and F. Wu. Digital twin-enabled dynamic scheduling with preventive maintenance using a double-layer q-learning algorithm. *Computers and Operations Research*, 144:105823, 2022. DOI: 10.1016/j.cor.2022.105823. URL: https://doi.org/10.1016/j.cor.2022.105823 (cited on pages 16, 25).

[YZC+19] T. Yu, C. Zhu, Q. Chang, and J. Wang. Imperfect corrective maintenance scheduling for energy efficient manufacturing systems through online task allocation method. *Journal of Manufacturing Systems*, 53:282–290, 2019. DOI: 10.1016/j.jmsy.2019.11.002. URL: https://doi.org/10.1016/j.jmsy.2019.11.002 (cited on page 30).

[ZAO+20] A. A. M. Zahir, S. S. N. Alhady, W. A. F. W. Othman, A. A. A. Wahab, and M. F. Ahmad. Objective functions modification of ga optimized pid controller for brushed dc motor. *International Journal of Electrical and Computer Engineering*, 10(3):2426–2433, 2020. DOI: 10.11591/ijece.v10i3.pp2426-2433. URL: https://doi.org/10.11591/ijece.v10i3.pp2426-2433 (cited on page 28).

[ZdCd+20] Tiago Zonta, Cristiano André da Costa, Rodrigo da Rosa Righi, Miromar José de Lima, Eduardo Silveira da Trindade, and Guann Pyng Li. Predictive maintenance in the Industry 4.0: A systematic literature review. *Comput Ind Eng*, 150:106889, December 2020. ISSN: 03608352. DOI: 10.1016/j.cie.2020.106889 (cited on page 8).

[ZTC+23] Z. Zhuang, H. Tao, Y. Chen, V. Stojanovic, and W. Paszke. 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, 2023. DOI: 10.1109/TSMC.2022.3225381. URL: https://doi.org/10.1109/TSMC.2022.3225381 (cited on page 28).

[ZTZ+21] Tong Zhou, Dunbing Tang, Haihua Zhu, and Zequn Zhang. Multi-agent reinforcement learning for online scheduling in smart factories. *Robotics and Computer-Integrated Manufacturing*, 72:102202, 2021 (cited on page 61).

[ZZ17] Ding Zhang and Yingjie Zhang. Dynamic decision-making for reliability and maintenance analysis of manufacturing systems based on failure effects. *Enterprise Information Systems*, 11(8):1228–1242, 2017 (cited on page 4).