AI-based maintenance planning for multi-component systems considering different kinds of dependencies

Van-Thai Nguyen

Ecole Doctorale IAEM Lorraine

CRAN, CNRS UMR 7039

Thèse

Présentée et soutenue publiquement pour l'obtention du titre de

DOCTEUR DE l'UNIVERSITE DE LORRAINE

Mention : Automatique, Traitement du Signal et des Images, Génie Informatique

par

Van-Thai NGUYEN

AI-based maintenance planning for multi-component systems considering different kinds of dependencies

12 Juin 2023

Membres du jury :

Rapporteurs :	Anne BARROS	PU	CentraleSupelec
	Chi-Guhn LEE	PU	University of Toronto
Examinateurs :	Antoine GRALL	PU	Université de technologie de Troyes (Président du jury)
	TP Khanh NGUYEN	MdC	Université de Toulouse
	Phuc DO	MdC, HDR	Université de Lorraine (Directeur de thèse)
	Alexandre VOISIN	MdC	Université de Lorraine (Co-encadrement)

Acknowledgment

First and foremost, I would like to express my greatest thanks to my supervisor Prof. Phuc DO for his help, support and fruitful discussions that made this thesis possible.

My special gratitude is given to my co-supervisor Dr. Alexandre VOISIN for his helpful and constructive comments, guidance and continuous support.

I wish to express my gratitude to the European commission for the financial support over this period. I also want to send my thanks to all the partners of AI-PROFICIENT project for their support and for the nice time we spent together through several meetings.

Finally, I would like to express my great gratefulness to my family and friends for their consistent stimulation of my dream pursuit.

Abstract

Maintenance planning for systems consisting of multiple components has still been a challenging problem. Particularly, mathematically describing dependencies between components is usually a complicated task, however, omitting component dependencies in maintenance modeling might result in suboptimal plans. Moreover, the number of maintenance decision variables needed to be optimized increases rapidly in the number of components, causing computational expense for optimization algorithms.

To face these issues, this PhD aims to propose an artificial-intelligence-based maintenance optimization approach allowing to consider different kinds of dependencies between components (i.e., economic, stochastic, and structural dependence). Particularly, the maintenance approach integrates a deep maintenance cost model, that allows to compute maintenance costs at system level without requiring individual costs at component level (e.g., setup costs, labor costs and costs of maintaining each component), into the framework of multi-agent deep reinforcement learning, which can be applicable for large sequential decision-making problems, to optimize maintenance decisions. Moreover, a novel degradation interaction model for discrete- state components is also developed and then integrated into the proposed maintenance approach. Numerical studied are conducted on multi-component systems with different configurations under different observability scenarios to investigate the performance, the advantage as well as limits of the proposed maintenance approach.

Keywords: maintenance decision-making, dependence modeling, multi-component systems, artificial intelligence, deep reinforcement learning, multi-agent systems.

Résumé

Le maintien en condition opérationnel de systèmes industriels reste un challenge important en regard des dépendances multiples entre composants (ex. dépendance économique, stochastique et structurelle) et du grand nombre de variables de décision en maintenance à optimiser.

Pour faire face à ce défi, cette thèse vise à proposer une approche d'optimisation de la maintenance basée sur l'intelligence artificielle permettant de prendre en compte différents types de dépendances entre composants. En particulier, l'approche de maintenance proposée intègre un modèle de prédiction basé sur des réseaux des neurones, pour l'estimation des coûts de maintenance au niveau du système sans avoir besoin des coûts individuels au niveau des composants, dans le cadre de l'apprentissage par renforcement profond multi-agents, qui peut être appliqué à la décision séquentielle de grande échelle, afin d'optimiser les décisions en maintenance. En outre, un nouveau modèle de dépendance d'états entre composants est également développé et ensuite intégré dans l'approche de maintenance proposée. De nombreux études numériques sont menées sur des systèmes avec différentes configurations sous différents scénarios d'observabilité pour étudier la performance et les avantages ainsi que des limites de l'approche de maintenance proposée.

Mots clés : Décision en maintenance, dépendance, systèmes à composants multiples, intelligence artificielle, apprentissage par renforcement profond, systèmes multi-agents.

Acronyms

AI	artificial intelligence
ANN	artificial neural network
CBM	condition-based maintenance
CM	corrective maintenance
CTDE	centralized training and decentralized execution
DCMAC	deep centralized multi-agent actor-critic
DDQN	double deep Q-network
DQN	deep Q-network
Dec-POMDP	decentralized partially observable Markov decision process
CM	corrective maintenance
DRL	deep reinforcement learning
IM	imperfect maintenance
GA	generic algorithm
GRU	gated recurrent unit
MADRL	multi-agent deep reinforcement learning
MDP	Markov decision process
MMDP	multi-agent Markov decision process
MPL	multi-layer perceptron
OEM	original equipment manufacturer
PM	preventive maintenance
POMDP	partially observable Markov decision process
RL	reinforcement learning
SD	stochastic dependence
VI	value iteration algorithm
WQMIX	weighted QMIX algorithm

Notations

N	number of components
N^{sub}	number of subsystems
$N^{sub,j}$	number of components of subsystem/type j
$N^{m,sys}$	number of maintained components
$N^{m,sub,j}$	number of maintained components of subsystem j
$\mathbb{I}^{dt}$	system downtime indicator
$\mathbb{I}^{m,sys}$	system maintenance indicator
$\mathbb{I}^{m,c,i}$	maintenance indicator of component i
$\mathbb{I}^{m,sub,j}$	maintenance indicator of subsystem j
$c^{ins,i}$	inspection cost of component i
$c^{s,sys}$	system setup cost
$c^{s,type,j}$	setup cost of component of type j
$c^{m,i}$	component-specific maintenance cost
c^{i}	cost of maintaining component i separately
$c^{r,i}$	cost of replacing component i by a new one
c^{sys}	maintenance cost at system level
c^{dt}	downtime cost
m^i	$m^i + 1$ is the number of health condition states of component i
s_k^i	state of component i before maintenance at timestep t_k
$\bar{s}_k^i$	state of component i after maintenance at timestep t_k
$\boldsymbol{s}_k$	system state before maintenance at timestep t_k
$\bar{\boldsymbol{s}}_k$	system state after maintenance at timestep t_k
a_k^i	maintenance action of component i at timestep t_k
$\boldsymbol{a}_k$	system maintenance action at timestep t_k
$\tilde{\boldsymbol{P}}^i$	inherent transition matrix of component i
$\boldsymbol{P}^i$	transition matrix of component i affected by stochastic dependence

τ^i	degradation interaction factor of component i
ζ^{ij}	constant quantifying the structure influence of component j on degradation nature of component i
α^j	constant describing the inherent degradation interaction characteristics of component j
ξ_{uv}^i	maximal change in the original transition probability from state u to v of component i
$\boldsymbol{Z}$	degradation interaction matrix
Q^i	action-value function of agent i
Ω^i	advantage function of agent i
$\boldsymbol{Q}$	joint action-value function
$\boldsymbol{Q}^{tot}$	factorized joint action-value function
$\boldsymbol{V}$	state-value function

Résumé de la thèse

De nombreux systèmes modernes tels que les lignes de production, les parcs éoliens, les flottes d'avions, etc., sont généralement considérés comme des systèmes à composants multiples soumis à la détérioration et au vieillissement en raison de l'utilisation et de l'exposition aux facteurs environnementaux, ce qui peut entraîner des dommages aux équipements, des problèmes de sécurité et l'indisponibilité imprévue des machines [1]. Par conséquent, la maintenance doit être effectuée pour maintenir ou rétablir ces systèmes dans un état de fonctionnement leur permettant d'accomplir les fonctions prévues et pour minimiser les temps d'arrêt imprévus [2–4].

Malgré le rôle important de la maintenance dans le maintien des systèmes en service, la littérature souligne que les coûts de maintenance représentent une part importante des coûts d'exploitation globaux. En particulier, les coûts de maintenance représentent 15% à 70% des coûts des biens produits dans les usines de fabrication et de production, et un tiers d'entre eux peuvent être dus à des opérations de maintenance inutiles ou mal exécutées [5]. Selon [6], les dépenses de maintenance sont l'une des composantes les plus importantes des coûts des parcs éoliens offshore. Plus précisément, le montant dépensé pour l'exploitation et la maintenance d'un parc éolien peut représenter plus de 25% du coût de son cycle de vie. Dans l'industrie chimique et de transformation, le personnel affecté aux opérations de maintenance peut représenter jusqu'à un tiers de la main-d'œuvre totale [7]. En outre, un contrat de maintenance pour une flotte d'avions s'étend souvent sur de nombreuses années et les coûts de maintenance cumulés peuvent dépasser des milliards de dollars, comme indiqué dans [8]. Il est donc nécessaire de gérer les coûts de maintenance de manière intelligente.

Dans cette optique, de nombreuses politiques de maintenance ont été proposées pour mieux programmer les opérations de maintenance, qui peuvent être classées en deux groupes principaux : la maintenance corrective et la maintenance préventive (CM et PM) [9]. Alors que la première consiste à réparer les machines défectueuses et est généralement associée à des coûts élevés en raison de défaillances soudaines, la seconde consiste à effectuer des opérations

de maintenance sur des machines en état de marche dans le but d'éviter les défaillances potentielles et de réduire les coûts d'immobilisation non planifiés [10, 11]. Les interventions de maintenance peuvent être programmées en fonction de l'âge ou de l'état de santé des machines. Par rapport à la première, la seconde, également connue sous le nom de maintenance conditionnelle (CBM), permet de prendre des décisions de maintenance basées sur l'état de santé réel des machines entretenues au lieu d'un calendrier fixe [10, 12, 13].

Presque toutes les approches CBM existantes considèrent les machines comme des systèmes à composant unique afin de faciliter la modélisation mathématique et les processus d'optimisation de la maintenance [14, 15]. Cependant, les politiques de maintenance qui en résultent peuvent ne pas être optimales si l'on considère des hypothèses réalistes, car les systèmes modernes sont souvent composés de multiples éléments et soumis à de multiples types de dépendance (par exemple, dépendance économique, stochastique et structurelle) [14, 16]. En outre, le nombre de variables de décision de maintenance à optimiser pour les systèmes à composants multiples augmente rapidement avec le nombre de composants, ce qui entraîne des calculs lourds pour les processus d'optimisation ou peut conduire à la découverte de solutions sous-optimales. Par conséquent, **l'objectif principal de ce doctorat est de planifier efficacement les interventions CBM pour les systèmes multi-composants en tenant compte de différents types de dépendances.**

Les avancées récentes dans le domaine de l'apprentissage par renforcement profond (DRL), plus précisément le DRL multi-agents (MADRL), ouvrent une nouvelle voie pour résoudre les problèmes de prise de décision en matière de maintenance des systèmes complexes [3, 11]. En particulier, le MADRL permet d'optimiser de manière efficace les décisions de maintenance séquentielles pour les systèmes multi-composants avec de grands espaces d'état et d'action [17]. En outre, il permet également d'adapter de manière flexible les politiques de maintenance à différents contextes d'observabilité [18, 19] (les décisions de maintenance peuvent être prises sur la base d'informations au niveau des composants ou du système). Cependant, l'application des algorithmes MADRL à l'optimisation des plans de maintenance pour les systèmes multi-composants se heurte encore à plusieurs difficultés liées à la con-

struction de modèles de coûts de maintenance du système utilisés pour définir la fonction de récompense, et de modèles de dépendance de dégradation des composants utilisés pour décrire la dynamique de transition probabiliste.

Tout d'abord, les modèles de coûts de maintenance des systèmes prennent généralement en compte la dépendance économique entre les composants maintenus, ce qui signifie que le coût de maintenance conjoint de plusieurs composants n'est pas égal au coût de maintenance de chacun d'entre eux séparément [16, 20]. Conventionnellement, des modèles de coûts explicites au niveau des composants, tels que les coûts d'installation, les coûts des pièces de rechange, les coûts de la main-d'œuvre de maintenance, etc. sont nécessaires pour construire le modèle de coûts au niveau du système [21, 22]. Cependant, dans la pratique, les actions de maintenance sont souvent regroupées pour être effectuées lors de chaque intervention de maintenance en raison de la dépendance économique des composants, ce qui fait que ces coûts individuels ne sont pas enregistrés séparément, mais que seul le coût total est documenté. Par conséquent, l'exigence de disponibilité de la collecte séparée des coûts individuels pour construire des modèles de coûts de maintenance au niveau du système pour les systèmes à plusieurs composants est moins pratique. C'est pourquoi la première question scientifique est de savoir *"comment développer une méthode permettant de calculer les coûts de maintenance au niveau du système qui puisse s'affranchir de la nécessité d'accéder aux coûts individuels liés à la maintenance au niveau des composants."*

Deuxièmement, la dynamique de transition du système du cadre MADRL est décrite par le modèle de dépendance de dégradation des composants dans le contexte de la planification de la maintenance, ou en d'autres termes, par le modèle de dépendance stochastique. Selon [23], la dépendance stochastique peut se produire en raison de la défaillance d'un composant qui cause un dommage unique ou augmente la vitesse de dégradation d'autres composants, ou lorsque les composants sont soumis à des défaillances en mode commun. Cependant, dans certains systèmes, la dépendance stochastique est déclenchée par les interactions de dégradation des composants au lieu d'attendre la défaillance complète d'un composant [24]. Dans la littérature, les approches de modélisation de ce type de dépendance stochastique ne

sont proposées que pour les composants à états continus et ne peuvent pas être directement appliquées aux composants à états discrets. Néanmoins, [25] souligne que dans de nombreuses applications pratiques, les composants et les systèmes doivent être modélisés par des modèles à états discrets pour des raisons économiques et/ou techniques. Par conséquent, la deuxième question scientifique est la suivante : *"comment modéliser la dépendance stochastique par le biais des interactions de dégradation dans les systèmes constitués de composants à état discret ?"*

Pour faire face aux questions scientifiques mentionnées ci-dessus, deux contributions principales sont fournies dans ce doctorat :

- **Proposition d'une approche d'optimisation de la maintenance basée sur le cadre de MADRL qui permet de se débarrasser de la nécessité d'accéder aux coûts individuels au niveau des composants dans le calcul des coûts de maintenance au niveau du système.**
- **Proposition d'un nouveau modèle pour décrire les interactions de dégradation dans les systèmes multi-composants basé sur les processus de Markov.**

En ce qui concerne les deux contributions principales, cette thèse est organisée comme suit :

- Le chapitre 1 présente une introduction aux systèmes multi-composants et à leurs caractéristiques, ainsi qu'au concept de CBM. En outre, les principaux problèmes liés à la planification de CBM pour les systèmes multi-composants sont également discutés, ce qui permet d'identifier les questions de recherche considérées dans ce doctorat.
- Le chapitre 2 présente la revue de la littérature relative aux questions de recherche identifiées dans la planification de la maintenance pour les systèmes constitués de composants multiples soumis à différents types de dépendances. En particulier, les approches de modélisation utilisées pour décrire la dépendance stochastique, économique et structurelle sont présentées. Les politiques de maintenance basées sur les seuils et la cartographie directe pour les systèmes à composants multiples sont également abordées. L'état de l'art sur ces aspects permet de mettre en évidence les deux questions scientifiques.

- Le chapitre 3 présente les concepts clés de MADRL qui implique le processus de prise de décision séquentielle de plusieurs agents vivant dans un environnement stochastique où les agents peuvent observer partiellement ou totalement les états réels de l'environnement. Les composants principaux de l'algorithme WQMIX [26], qui est utilisé pour optimiser les décisions de maintenance des systèmes étudiés dans cette thèse, sont également présentés.

- Le chapitre 4 est consacré à la présentation d'une approche d'optimisation de la maintenance basée sur l'IA proposée pour prendre en compte la première question scientifique. En particulier, un modèle de coût de maintenance au niveau du système est d'abord appris par un réseau neuronal profond à partir de données de maintenance conditionnelle dédiées, ce qui permet d'éviter la nécessité d'accéder aux coûts individuels au niveau des composants pour calculer les coûts de maintenance au niveau du système. Le modèle d'évolution de la dégradation du système considéré et le modèle de coût appris sont ensuite utilisés pour construire un environnement interactif pour des agents d'apprentissage par renforcement coopératif. Ensuite, les politiques de maintenance optimales sont apprises en laissant les agents d'apprentissage par renforcement coopératif interagir avec l'environnement construit.

- En lien avec la deuxième question scientifique, le chapitre 5 présente un nouveau modèle d'interaction état-dégradation basé sur les processus de Markov, dans le sens où la dégradation d'un composant peut accélérer la vitesse de détérioration d'autres composants. En outre, les composants des systèmes étudiés dans ce chapitre sont également supposés être soumis à la dépendance économique. Les politiques de maintenance de ces systèmes sont optimisées par l'approche proposée au chapitre 4, en tenant compte du fait que les décideurs en matière de maintenance ne peuvent accéder qu'à des informations au niveau des composants (données locales) lorsqu'ils prennent des décisions.

- Enfin, les conclusions et les perspectives tirées de ce doctorat sont présentées au chapitre 6.

Table of Contents

List of Figures

List of Tables

Introduction

Many modern systems such as production lines, wind farms, fleet of aircraft, etc., are in general considered as multi-component systems subjected to deterioration and aging due to usage and exposure to environmental factors which might lead to equipment damage, safety issues, and unplanned machine unavailability [1]. Therefore, maintenance has to be carried out to sustain these systems in, or restore it to, an operating state in which they can perform designated functions as well as to minimize unforeseen downtime [2–4].

Despite the important role of maintenance in keeping systems in service, it is pointed out in the literature that maintenance costs contribute to a significant portion of overall operating costs. In particular, maintenance costs represent $15\% - 70\%$ of the costs of goods produced in manufacturing and production plants, and a third of them might be caused by unnecessary or poorly executed maintenance operations [5]. According to [6], maintenance expenditures are one of the largest component costs for offshore wind farms. Specifically, the amount of money spent for operation and maintenance of a wind farm can account for over 25% of its life-cycle cost. In the process and chemical industry, the total amount of personnel related to maintenance operations can be up to a third of the total workforce [7]. Moreover, a maintenance contract for a fleet of aircraft often spans many years and the accumulative maintenance costs might exceed billions of dollars as specified in [8]. Therefore, it is necessary to manage maintenance costs intelligently.

With this in mind, many maintenance policies have been proposed for better scheduling maintenance operations which can be sorted into two primary groups: corrective and preventive maintenance (CM and PM) [9]. While the former repairs malfunctioned machines and is usually associated with high costs due to sudden failures, the later considers performing maintenance on functioning machines with the objective of avoiding potential failures to reduce unplanned downtime costs [10, 11]. PM interventions can be scheduled according to either age or health condition states of machines. In comparison with the former, the latter also known as condition-based maintenance (CBM) allows to make maintenance decisions

based on actual health condition of maintained machines instead of a fixed calendar [10, 12, 13].

Almost all exiting CBM approaches consider machines as single-component systems to facilitate mathematical modeling and maintenance optimization processes [14, 15]. However, the resulting maintenance policies might not be optimal when considering realistic hypothesis due to the fact that modern systems are often composed of multiple components and subjected to multiple dependency types (e.g., economic, stochastic and structural dependence) [14, 16]. Moreover, the number of maintenance decision variables needed to be optimized for multi-component systems increases rapidly as the number of components grows, which causes heavy computation for optimization processes or might lead to the finding of sub-optimal solutions. Therefore, **the main objective of this PhD is to effectively plan CBM interventions for multi-component systems considering different kinds of dependencies.**

Recent advances in the field of deep reinforcement learning (DRL), more precisely, multi-agent DRL (MADRL), open a new direction to solve maintenance decision-making problems of complex systems [3, 11]. In particular, MADRL allows to optimize sequential maintenance decisions for multi-component systems with both large state and action spaces in an effective way [17]. Moreover, it also allows to flexibly adapt maintenance policies to different observability settings [18, 19] (maintenance decisions can be made based on information at either component or system level). However, applying MADRL algorithms to optimize maintenance plans for multi-component systems still faces several challenges related to the construction of system maintenance cost models used to define the reward function, and of component degradation dependence models used to describe the probabilistic transition dynamic.

Firstly, system maintenance cost models usually take into account the economic dependence between maintained components which means that the joint maintenance cost of several components is not equal to the cost of maintaining them separately [16, 20]. Conventionally, explicit cost models at component level such as setup costs, spare part costs, maintenance labor costs, etc. are required to build the cost model at system level [21, 22]. However, main-

tenance actions are often grouped to carry out in each maintenance intervention in practice due to the component economic dependence, which leads to the fact that such individual costs are not recorded separately, instead, only total cost is documented. Therefore, the availability requirement of separately collecting individual costs to construct maintenance cost models at system level for multi-component systems is less practical. For this reason, the first scientific issue is *"how to develop a method allowing to compute maintenance costs at system level that can get rid of the demand of accessing individual maintenance-related costs at component level?"*

Secondly, the system transition dynamic of MADRL framework is described through the component degradation dependence model in the context of maintenance planning, or in other words, via the stochastic dependence model. According to [23], the stochastic dependence can occur due to the failure of a component which causes one-time damage for or increases degradation speed of other components, or when components are subjected to common-mode failures. However, the stochastic dependence in some systems is triggered by component degradation interactions instead of waiting until component's complete failure to occur [24]. In the literature, modeling approaches of this kind of stochastic dependence are only proposed for components with continuous states and cannot be directly applied for discrete-state components. Nevertheless, it is pointed out in [25] that in many practical applications, components and systems should be modelized by discrete-state models due to economic and/or technical reasons. Therefore, the second scientific issue is *"how to model the stochastic dependence through degradation interactions in systems consisting of discrete-state components?"*

To face the scientific issues mentioned above, two main contributions are provided in this PhD:

- **Proposal of a maintenance optimization approach based on the framework of MADRL that allows to get rid of the demand of accessing individual costs at component level in the computation of maintenance costs at system level.**
- **Proposal of a novel model for describing degradation interactions in multi-**

component systems based on Markov processes.

In regard to the two main contributions, this thesis is organized as follows:

- Chapter 1 presents an introduction to multi-component systems and their characteristics as well as the concept of CBM. Moreover, main problems in CBM planning for multi-component systems are also discussed through which the research questions considered in this PhD is identified.
- Chapter 2 presents the literature review related to the identified research questions in maintenance planning for systems consisting multiple components subjected to different kinds of dependencies. Specifically, modeling approaches used for describing stochastic, economic and structural dependence are presented. Threshold-based and direct-mapping maintenance policies for multi-component systems are also discussed. Through the state-of-the-art on these aspects, the two scientific issues are highlighted.
- Chapter 3 presents the key concepts of MADRL which involves the sequential decision-making process of multiple agents inhabiting in a stochastic environment where the agents can either partially or fully observe true states of the environment. The core components of WQMIX algorithm [26], which is used to optimize maintenance decisions of the systems studied in this thesis, is also presented.
- Chapter 4 is devoted to the presentation of an AI-based maintenance optimization approach proposed to take the first scientific issue into consideration. Particularly, a maintenance cost model at system level is first learned by a deep neural network from a dedicated condition monitoring data that allows avoiding the requirement of accessing individual costs at component level to compute system-level maintenance costs. The degradation evolution model of the system under consideration and the learned cost model are then used to construct an interactive environment for cooperative reinforcement learning agents. After that, optimal maintenance policies are learned by letting cooperative reinforcement learning agents interact with the constructed environment.
- In link with the second scientific issue, Chapter 5 presents a novel state-rate degrada-

tion interaction model based on Markov processes in the sense that the degradation of a component can accelerate the deterioration speed of other components. Moreover, the components of the systems studied in this chapter are also supposed to be subjected to the economic dependence. Maintenance policies of these systems are optimized by the proposed approach presented in Chapter 4 taking into account the case where maintenance decision-makers can only access to information at component level (local data) when making decisions.

- Finally, conclusions and perspectives drawn from this PhD are presented in Chapter 6.

Chapter 1

An overview of maintenance planning for multi-component systems

1.1 Introduction

Modern systems are generally multi-component systems subjected to deterioration and aging which might cause safety issues and unexpected unavailability. Hence, maintenance has to be carried out to sustain these systems in, or restore them to, an operating states in which they can perform designated functions [2]. A lot of maintenance strategies has been developed over time to optimally plan maintenance activities, among which CBM nowadays becomes sophisticated and is a popular approach for maintenance decision-making and optimization thanks to recent advances in Industry 4.0 technologies providing massive useful health condition data [27–29]. However, determining optimal CBM policies for multi-component systems faces many challenges concerning maintenance modeling and maintenance decision optimization.

In order to help determine scientific gaps, this chapter aims to give an overview of multi-component systems and to present main problems in their maintenance planning process. Specifically, Section 1.2 presents an introduction to multi-component systems as well as their characteristics. Three important steps of CBM including condition monitoring, maintenance modeling and maintenance decision optimization are presented in Section 1.3. The next section is devoted to outline difficulties in maintenance planning optimization for systems consisting of dependent components from which the research questions considered in this PhD are identified. Finally, Section 1.5 concludes the chapter.

1.2 Multi-component systems

Maintenance planning research at early state used to consider machines or systems being maintained as single-component systems to facilitate mathematical modeling and reliability analysis and optimization processes. The term "single-component" or "single-unit" is employed to depict a system in which there is only one component, or in more broad sense, to describe a system where there is a unique critical component that is supposed to be able to represent the entire system [30].

However, this simplification does not seem so relevant today since engineering systems are getting more and more complex, and usually consist of multiple components due to increasing demand in higher performance and safety [24]. Moreover, considering maintained systems as single-component systems does not allow taking into account component interactions or dependencies (i.e., economic, stochastic and structural dependence) in maintenance scheduling that could result in suboptimal maintenance plans [4, 31, 32]. Therefore, maintenance planning for multi-component systems have been received a lot of attention in the literature since last few decades.

In general, a multi-component system might have a hierarchical structure which depends on its scale [16]. For instance, a system may comprise of subsystems which consist of smaller subsystems where there is one or several components. Despite the complexity of determining a suitable hierarchical structure for a system, a component is often considered as a part of a system that is subjected to maintenance interventions and can not be further divided into sub-components that are individually subjected to any maintenance intervention [1].

A production line can be considered as a system where its subsystems are connected in series from a reliability block diagram point of view, which means that if one of the subsystems is not functioning, the whole system is not operating. Moreover, in each subsystem, components might have series, parallel or mixed structure. An illustration of a series-parallel production line is depicted in Figure 1.1. Particularly, the production line system contains 8 components which are grouped into 4 subsystems. Component 1 is considered as the first subsystem. The

next subsystem is composed of component 2 and 3. Component 4, 5 and 6 together form the third subsystem. The last one consists of the remaining components. Other examples of multi-component systems are motor engines, machine tools, gear boxes and industrial cool box systems [33–35].

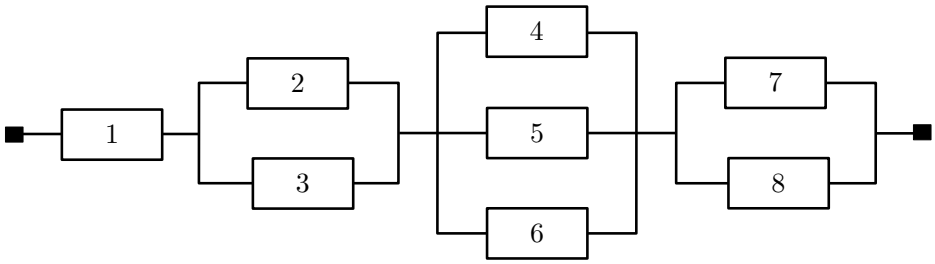


Figure 1.1: Reliability block diagram of a series-parallel production line.

1.3 Condition-based maintenance

Maintenance interventions can be carried out in either corrective and preventive maintenance manner resulting in CM and PM strategies [9]. CM is also known as breakdown or run-to-failure maintenance, which repairs malfunctioned machines and is usually associated with high cost due to unexpected production losses [10, 36]. On the contrary, PM interventions are implemented on functioning machines to avoid failures leading to reduce unplanned downtime costs [11]. There exists in the literature two main kinds of PM actions: time-based maintenance (TBM) and CBM.

According to TBM, maintenance interventions are carried out at regular time, for example, every 500 hours, based on either experiences of repairmen or original equipment manufacturer (OEM) recommendations which suffer from several drawbacks. Particularly, PM based on experience becomes problematic when the experienced person leaves the organization. In addition, it's possible that these individuals might not always on duty in the production line to address maintenance issues [37]. Performing maintenance based on OEM advice are typically not correct due to the variations between actual operating circumstances and those used for lab testing as analyzed in [10].

CBM is a recent maintenance technique allowing to make preventive maintenance decisions based on machines' actual health condition [38] which is motivated by the fact that 99% of equipment failures are preceded by certain signs, conditions, or indications that a failure is going to occur [37]. Therefore, it is necessary to perform CBM in reality for better equipment health management, lower life cycle cost, catastrophic failure avoidance [10]. Moreover, thanks to significant advances achieved in sensing technology recently allowing to collect rich degradation measurement information, CBM has become a popular approach for maintenance decision-making and optimization [12, 13, 39], and is the main focus of this PhD.

In general, there are three main steps in CBM which are condition monitoring, maintenance modeling and maintenance decision optimization as illustrated in Figure 1.2 where the last two steps can be considered as maintenance planning process [10, 14].

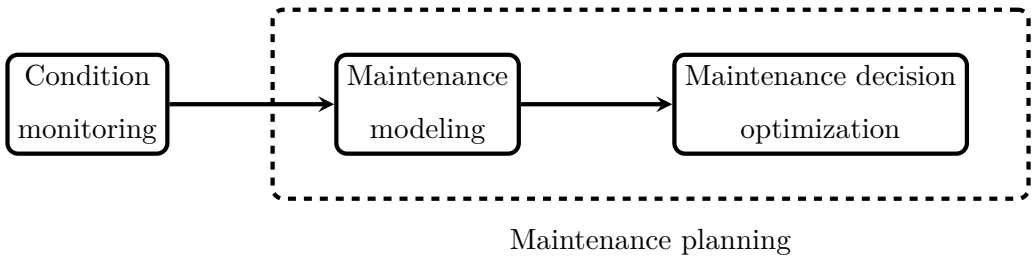


Figure 1.2: Three important steps of condition-based maintenance optimization.

The objective of condition monitoring is to monitor the lifetime (age) of components/systems through their operating condition, which can be measured based on various physical signals, such as vibration, temperature, accumulative wear, lubricating oil, and noise levels [10]. Condition monitoring can be implemented in either continuous or discrete manner. While the former reveals equipment's health condition in real time based on sensors integrated in the equipment, the later measures equipment's condition through inspections carried out at certain intervals such as every hour or week (periodical inspection), or based on the requirement of maintenance policies (non-periodical inspection).

The second step in CBM is maintenance modeling which involves mathematically describing

components' degradation process as well as interactions or dependencies in multi-component systems. The degradation process of a component is usually a random process which has either continuous and discrete form depending on situations under consideration. Particularly, continuous degradation models (e.g., Wiener, gamma and inverse Gaussian processes) are suitable for describing gradual deterioration over time such as wear, fatigue, corrosion, crack growth, erosion, consumption, creep, swell, etc. [40]. On the contrary, the discrete ones are used in the case where precise measurements of component degradation are difficult to obtain, or it is sometimes unnecessary to work on continuous values from an engineering practical point of view [14]. An example of discrete stochastic processes used to model degradation is Markov decision processes (MDPs) which are frequently employed to characterize the degradation process of civil engineering structures [41–43]. Another step of maintenance modeling is to characterize component dependencies including economic, stochastic and structural dependence. This step is also of importance because it affects the degradation process of the components as well as the quality of optimized maintenance policies. More detail about component dependence modeling will be discussed in Section 1.4.1.

The next step after obtaining a maintenance model of the system under consideration is to optimize maintenance decision variables, which can be either preventive maintenance thresholds or maintenance actions directly, according to some objective functions typically related to maintenance costs and production losses [18, 32]. Particularly, maintenance decision optimization is usually associated with the problem of balancing the trade-off between maintenance frequency and unplanned downtime cost in the sense that if maintenance actions are carried out more often maintenance costs might be high, or if maintenance operations are conducted less frequently, system breakdowns are more prone to occur leading to negative consequences [23]. Thus, a good maintenance plan could result in substantial advantages for the competitiveness of industrial enterprises (e.g., increasing system availability, benefits for safety management, reducing maintenance costs, improving product quality) [30].

Despite the promising benefit of CBM, there still exits several challenges in realizing this kind of maintenance strategy for multi-component systems. These challenges are discussed

in the following section.

1.4 Main problems in maintenance planning for multi-component systems

Maintenance planning for systems consisting of multiple components has been still a challenging problem. Particularly, maintenance modeling is usually a time-consuming and complicated task when component dependencies are taken into account. Moreover, the number of maintenance decision variables needed to be optimized increases rapidly in the number of components which makes optimization processes computationally expensive. The two following subsections are devoted to present these two main problems in maintenance planning for multi-component systems.

1.4.1 Maintenance modeling problems

The main difficulty in maintenance modeling for a multi-component system is how to correctly describe interactions inside the system, or in other words, dependencies between components, which significantly influence the degradation process of the components as well as its optimal maintenance decision-making strategy. Broadly, component dependencies can be divided into three main groups: *stochastic, economic, and structural dependence* [31, 32, 44, 45].

The stochastic dependence arises when component failures/degradation processes have interactions, or in other words, the failure/degradation process of a component influences other functioning components' lifetime distribution [16]. This kind of component dependency can be further classified into two classes. The first one is the *failure-based stochastic dependence* meaning that the stochastic dependence between components in a system is triggered by the complete failure of a component, or the components are subjected to common-mode failures [23]. For example, the stoppage of a pump in a pumping system that shares a common load will increase load on other working pumps. As a result, these functioning pumps will deteriorate faster. The second one is the *degradation-based stochastic dependence* which refers that

the component stochastic dependence is not necessarily triggered by component's complete failure but by component' degradation [24]. For example, the wear of a gear in a gearbox system can accelerate the degradation speed of its connected gears.

The economic dependence means that the joint maintenance cost of several components is not equal to the cost of maintaining them separately [20]. Particularly, combining maintenance on multiple components might be either cheaper *(positive economic dependence)* or more expensive *(negative economic dependence)* than maintaining them individually [23]. The positive economic dependence is often represented through the saving of setup costs, for example, cost of sending all members of a maintenance team to the site once is obviously less expensive than that of sending each of them separately. From a practical point of view, setup costs can be viewed as costs of ordering spare parts, shutting down the system for maintenance, etc. [32], and can be shared in a hierarchical structure [46]. On the contrary to the positive economic dependence, the negative one means that maintaining components simultaneously costs more than maintaining components individually, which is caused by safety requirements, manpower restrictions and production losses [16]. For instance, multiple maintenance workers operating in a limited space will start blocking and irritating each other and the probability of occurring human errors increases as a result. Another example is that a company might need to hire additional labor and/or buy new tools to be able to maintain components in group, which may be much costly. Maintaining several components together can also lead to the shutdown of the system, thus it is necessary to carefully plan grouping maintenance to avoid system unavailability [47].

The structural dependence exists when components in a system are structurally or functionally related to each other. There are two main kinds of structural dependence [23]: *technical and performance dependence*. The first one means that maintenance on certain components can either require or prohibit maintenance on other components [35]. For example, the cassette and chain of a bicycle indeed form a union, and the replacement of one requires the replacement of the other. The second kind of structural dependence refers that the overall system performance is dependent on the performance of the components as well as their

configuration within the system (series, parallel or mixed structure) that strongly affects the structure of maintenance policies being used [23]. For instance, maintenance policies used for a series system should perform maintenance on components at relatively early state to prevent downtime or to reduce unavailability costs, and should take into consideration the stoppage of a component as an opportunity to perform maintenance on certain components to save setup costs.

Almost all existing maintenance models in the literature consider only one specific kind of component dependencies since integrating more than one makes the modeling process more complicated [16, 23]. However, different dependency types in practice exist in many systems. For example, a gearbox system is studied in [20], which suffers from both economic and stochastic dependence. Particularly, the stochastic dependence is such that the wearing rate of each component depends not only on its own wearing level but also on the wearing state of other components. The economic dependence arises from shared setup costs. Moreover, omitting component dependencies in maintenance modeling might result in suboptimal maintenance plans which can cause huge consequences. For instance, the stochastic dependence in a multi-component system often lead to the fact that components tend to reach failure states faster and maintenance should be carried out earlier to prevent failures. Hence, if it is not integrated in the maintenance model that is then used in maintenance decision-making optimization process, optimized maintenance polices might be suboptimal and can lead to high system unavailability due to unexpected component failures. *Therefore, it is necessary to model and incorporate different kinds of component dependencies into maintenance models of multi-component systems.*

1.4.2 Maintenance decision optimization challenges

The second step in maintenance planning is to optimize maintenance decision variables which depends on the type of maintenance policy being used. Particularly, CBM policies can be divided into two main groups, namely, direct mapping and threshold-based policy. While the former maps directly from component degradation measurements to maintenance actions, the

later first compares component degradation states to predefined thresholds, and then choose maintenance actions accordingly. As a result, maintenance actions are decision variables needed to be optimized for direct-mapping policies, whereas thresholds which also known as control limits in the literature are decision variables for the optimization process of threshold-based maintenance policies.

Conventional threshold-based maintenance polices proposed for multi-component systems originate from the ones designed for single-component system requiring the optimization of N preventive thresholds for systems consisting of N components [14, 15]. However, these maintenance policies are suboptimal because they do not take component dependencies into account. To address this issue, many maintenance policies considering multiple thresholds per component are proposed for better maintenance planning, however, requiring more decision variables need to be optimized [48]. As shown in [20], an opportunistic maintenance policy designed for a two-component system considering only replacement actions requires four decision variables: two preventive thresholds and two opportunistic thresholds. It should be noted that optimal thresholds can be obtained by performing a exhaustive search on fine grid over their value space if the system is small. However, tailored heuristic search methods (e.g., genetic, ant algorithms and particle swarm optimization) are necessarily employed to optimize maintenance thresholds if the size of the system under consideration is relatively large [24, 48]. Moreover, it is worth noting that such heuristic search algorithms do not guarantee the finding of global optimal solutions, and thus are often needed to run several times to get the best results.

Optimizing direct-mapping maintenance policies in the simplest case can be viewed as solving MDPs which model the sequential decision making process of an agent in an uncertain environment [49, 50]. Finding optimal solutions for a MDP can be done by using conventional algorithms such as dynamic programming and tabular reinforcement learning if its size is relatively small. However, solving a MDP becomes difficult if its size is large due to the curse of dimensionality. In the context of maintenance planning, this means that the state and action space of a multi-component system grows exponentially in the number of components

causing computationally expensive for optimization algorithms [17]. For example, considering a system consisting of N components in which a component i ($i = 1, 2, \ldots, N$) has $m^i + 1$ discrete health condition states, the state space size is $\prod_{i=1}^{N}(m^i + 1)$, and the action space size is 2^N if only replacement actions are considered (i.e., at each decision epoch, a component is decided to be replaced or not). It should be noted that the action space of the mentioned system gets even larger once imperfect maintenance is taken into account in maintenance planning [18]. For instance, if "minor repair" and "major repair" action are taken into consideration, the number of maintenance actions needed to be optimized is 4^N.

Moreover, deciding a maintenance action to implement gets harder if the degradation state of the system under consideration is difficult to obtain due to the imperfection of condition monitoring processes [42, 43]. Maintenance decision making problem of multi-component systems becomes even more difficult if there are multiple maintenance decision-makers involved who have to cooperate with each other to make maintenance decisions for components/subsystems in a decentralized way [19] (the maintenance decision-maker of a component/subsystem know neither degradation states of other components/subsystems nor maintenance actions chosen by other decision-makers while making maintenance decisions). In particular, gathering and transmitting system-wide data for some large-scale industrial systems may be inefficient or even impossible [51]. Thus, selecting maintenance actions for each machine based on local data (local observability) is appropriate for a such complex system. However, optimizing performance at component/machine level might not result in the optimal collective objective at system level due to the dependencies inside the system [17].

Based on these analysis above, it is clear that effectively optimizing CBM decisions for large multi-component systems considering component dependencies and local observability is a challenging problem.

1.5 Conclusions

This chapter presented an overview of CBM planning for multi-component systems which consists of two main steps: maintenance modeling and maintenance decision optimization. The problem in maintenance modeling is how to correctly describe the dependencies between components including the economic, stochastic and structural dependence. Omitting component dependencies in maintenance modeling might result in suboptimal maintenance plans leading to high maintenance costs. The curse of dimensionality is the main issue in maintenance decision optimization for systems composed of multiple components, which means that the number of maintenance decision variables needed to be optimized grows exponentially in the number of components. Moreover, deciding whether it is necessary to maintain a component is more difficult if information at system level is not available at decision time. Based on the analysis of these difficulties mentioned above, the research questions considered in this PhD are:

- *How to model and incorporate different kinds of dependencies into maintenance models of multi-component systems?*
- *How to effectively optimize CBM decisions for large multi-component systems taking into account component dependencies and local observability?*

In order to identify scientific gaps, a review of the state-of-the-art related to these two challenges is developed in the following chapter.

Chapter 2

Dependence modeling and maintenance decision-making for multi-component systems: State-of-the-art

2.1 Introduction

The objective of this chapter is to overview the existing works related to the research questions identified in previous chapter. Particularly, the state-of-the-art is developed with regards to maintenance planning approaches for multi-component systems taking into consideration component dependencies, based on which the scientific issues to be attacked are highlighted.

Chapter 2 is structured as follows. Section 2.2 are devoted to present a survey on dependence modeling methods for multi-component systems considering stochastic, economic and structure dependence. The state-of-the-art on maintenance decision optimization is provided in Section 2.3. Conclusions are presented in the last section.

2.2 Component dependencies

2.2.1 Economic dependence

Among three kinds of component dependencies, the economic dependence is most commonly considered in maintenance planning for multi-component systems, which simply implies that the joint maintenance cost of several components is not equal to the total cost of maintaining them separately [20]. The economic dependence can be further sorted into the positive and negative economic dependence based on their impact on the total maintenance cost

(maintenance cost at system level) [16, 23].

The positive economic dependence means that combining maintenance on multiple components is cheaper than maintaining them separately, which is often represented through the saving of setup costs. From a practical point of view, setup costs can be viewed as costs of ordering spare part, shutting down the system for maintenance, etc. [32]. The fixed setup cost model, which means that the setup cost for a group of maintenance activities that are executed together has to pay only once, is commonly considered in the literature [21]. Indeed, this kind of setup cost model is independent of the system structure, the type of maintenance that is performed, and time [23]. For example, the cost induced by crew traveling, scaffolding, etc. can be assumed to be the same for all maintenance activities. However, the setup cost sharing might have hierarchical structure when a system contains of several component types or subsystems. Particularly, two levels of setup costs are considered in [52], which are system setup cost caused by, for instance, administrative handling or transportation of spare parts, and component-type setup cost originated from the requirement of repairman skills or specific tools. If several maintenance actions are grouped, the system setup cost has to paid only once. Similarly, the component-type setup costs are charged only once if some components of the same type are maintained together. Despite the popularity of the assumption that the setup cost is not a function of time, it is time-dependent in some cases in reality. For example, the maintenance setup costs of a hydro-generating unit in a deregulated power system might be time-dependent because of fluctuations in the monthly electricity price as analyzed in [53].

On the contrary to the positive economic dependence, the negative one means that maintaining components simultaneously costs more than maintaining them individually, due to safety requirements, manpower restrictions and redundancy/production losses [16]. For instance, multiple maintenance workers operating in a limited space will start blocking and irritating each other and the probability of occurring human errors increases as a result. Another example in which the negative economic dependence can occur is when a company might needs to hire additional labor and/or buy new tools to be able to maintain components in

group. Maintenance cost may increase more than linearly with the number of maintenance activities and often represented by convex function of the number of components that receive maintenance [23].

It is important to note that the main objective of mathematically describing the economic dependence in a multi-component system is to obtain the cost model of maintaining several components simultaneously, in other words, to construct the maintenance cost model at system level. Traditionally, explicit cost models at component level (e.g., setup costs, spare part costs, maintenance labor costs) are required to build the cost model at system level [21, 22]. From a practical point of view, maintenance actions are often grouped in each maintenance intervention due to the component economic dependence, which leads to the fact that such individual costs are not recorded separately, instead, only total cost is documented. Therefore, the availability requirement of separately collecting individual maintenance costs to construct the cost model at system level is not realistic. For this reason, the first scientific issue is *"how to develop a method allowing to compute maintenance costs at system level that can get rid of the demand of accessing individual maintenance-related costs at component level?"*

2.2.2 Stochastic dependence

The stochastic dependence arises once the failure/degradation process of a component influences other functioning components' lifetime distribution. A summary of different kinds of stochastic dependence is presented in Figure 2.1.

Particularly, the failure-based stochastic dependence including failure-induced damage, load sharing, and common-mode failure have dominated the number of studies in this research area according to [23, 24]. The failure-induced damage means that the failure of one component can cause a major, one-time damage to other components, leading to an immediate increase in the deterioration level or even an immediate failure of these components [54–56]. For example, the over-wear of the break pads cause serious damage to the disc rotor in a breaking system [54]. The stochastic dependence through load sharing occurs in the case

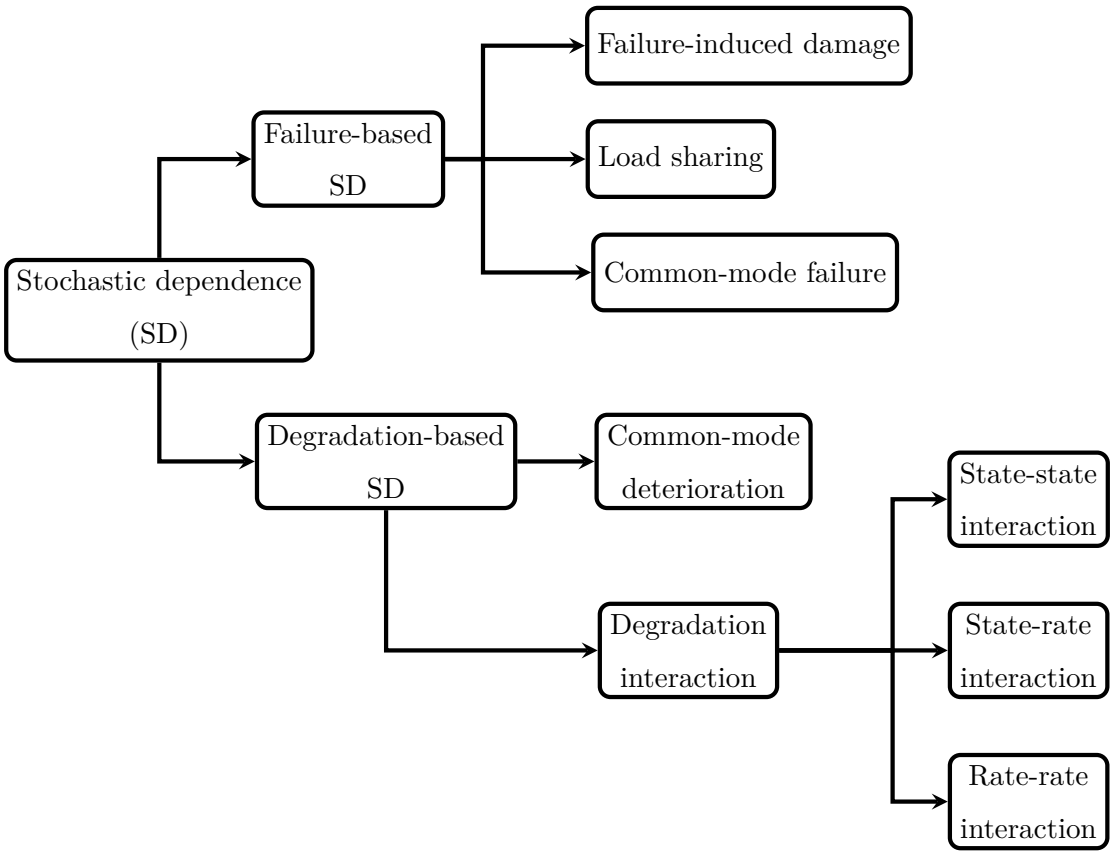


Figure 2.1: Classification of stochastic dependence.

where some components in a system share a common load [57–60]. Particularly, the system keeps functioning if one component suddenly stops working, but the remaining components structurally need to work harder to realize the same output level. As a result, these working components will degrade faster. In reality, this kind of failure dependence can be found in a set of pumps that are employed to distribute a certain amount of gas. The common-mode failure refers that several components can fail simultaneously due to similar working conditions [23, 61]. For instance, components' failure times in a system of stents implanted in human arteries are probabilistically dependent due to the exposure of stents to the same shocks [62].

From a practical point of view, it can be seen that the stochastic dependence in some systems is triggered by component's degradation instead of waiting until component's complete failure

to occur. This kind of component stochastic dependence can be divided into: common-mode deterioration and degradation interaction. The common-mode deterioration means that an increase in degradation of one component is associated with an increase in degradation of other components [20]. The degradation correlation between components is usually characterized by a joint probability distribution. Particularly, bivariate non-decreasing Lévy processes are used in [63, 64] to model the degradation correlation of two-component systems. Multivariate copula modeling is also employed to model this kind of stochastic dependence as done in [65–67]. The degradation interaction is a new type of stochastic dependence which considers the influence of degradation state/rate of a component on other components' individual degradation process. Specifically, the stochastic dependence through state-state interactions, which means that the increases in degradation level of a component is a function of its own state as well as other components' state, is studied in [68] for an industrial cool box. The authors of this paper then extended their research by characterizing the stochastic dependence through state-rate interaction [69], which means degradation levels of a component accelerate the degradation process of its dependent components. This kind of degradation interaction is also studied in [70] for a networked system using systems of continuous stochastic differential equations, and in [20] for a gearbox system whose components' degradation process are modeled by gamma processes. More recently, the component stochastic dependence through rate-rate interactions investigating how the degradation acceleration of one component affects the degradation rate of other components in a continuous way over time is developed in [24] for general multi-component systems using Wiener processes.

It can be noticed that the degradation interaction models developed in these above mentioned articles are only proposed for components with continuous states and cannot be directly applied for discrete-state components. Nevertheless, it is pointed out in [25] that in many practical applications, components and system should be modelized by discrete-state models due to economic and/or technical reasons. Firstly, considering only a few discrete deterioration states allows the decision-maker to have a more synthetic view on the system state as well as to have a simpler maintenance decision-making process. Secondly, a continuous condition monitoring program is sometime infeasible or very costly. For instance, a continuous

monitoring program of a remote pump station is often considered to be impractical due to problems related to wireless data transmission. Moreover, it would be extremely expensive to have an online monitoring system in place for every machinery component of a large complex engineering system (e.g., a water supply system) [78]. *Therefore, there is still room for modeling the degradation interaction in systems consisting of multi-state components which degrade discretely over time. (the second scientific issue)*

2.2.3 Structural dependence

The structural dependence implies that some or all components in a system are structurally connected to each other. According to [23], there are two primary types of structural dependence, namely, *technical and performance dependence*.

The technical structural dependence means that maintenance on a component can either require or prohibit maintenance on other components due technical system configuration that can cause maintenance or usage restrictions. For instance, it is required that the tires of an aircraft have the same thickness, and they thus should be replaced jointly [23]. Moreover, approaching a component that needs to be maintained might require the disassembly of other components that are blocking the access [35]. For example, in an "automobile engine and power transmission system", several steps must be followed in order to be able to maintain a shaft in the gearbox. Firstly, the gearbox unit is necessarily separated from the system, which can be done by a sequence of separating the engine, then the clutch. After that, the housing case of the gearbox needs to be opened and all the bearings and gears attached to the shaft are disassembled in order to get the shaft for maintenance [34]. The precedence relation governing the disassembly order of components in a system can be mathematically described by a directed graph which uses nodes to represent components or sub-systems, directed and undirected lines to present the component disassembly precedence [34, 35]. It is interesting to see in the literature that the maintenance constrains between components in some cases can be beneficial. Particularly, a multi-level opportunistic predictive maintenance approach is proposed in [48] for a convey system to take advantage of both economic and structural

dependence between components. Indeed, the disassembly of a component for maintenance might be considered as a good chance to also maintain the components in its disassembly path to reduce downtime costs.

The second type of structural dependence is the performance dependence referring that the overall system performance is dependent on the performance of the components as well as the way they are connected that strongly affects the structure of maintenance policies being used [23]. In a series system, the system fails if at least one of its component fails. Therefore, maintenance policies used for series systems should perform maintenance on components at relatively early states to prevent downtime, and should take into consideration the stoppage of a component as an opportunity to perform maintenance on certain components to reduce unavailability costs [71–75]. In cases where systems have parallel configuration which means that the system keep operating if at least one component is still functioning, maintenance actions can thus be performed at a relatively late stage [67, 76, 77] . In practice, components in a system can be connected according to various structure configurations from a reliability block diagram point of view, varying from a simple series system to a complex combination of components in parallel and series [63, 73, 78, 79] . The more complex the structure of systems being maintained is, the more difficult to determine an appropriate maintenance policy for these systems [80–82].

2.3 Maintenance optimization

In the context of CBM, the aim of maintenance optimization is to optimize maintenance policies which can be classified into two broad groups: threshold-based and direct-mapping maintenance policy [16]. The former imposes a predefined structure to maintenance policies via a relative small number of thresholds which are then needed to be tuned to the best values. On the contrary, the later does not impose any restriction on maintenance policies, instead its objective is to find a direct mapping from components' health condition to optimal maintenance actions. The following subsections are devoted to give a survey on these two types of CBM policies.

2.3.1 Threshold-based maintenance policies

This kind of maintenance policy is very common in the literature which is specified by multiple thresholds per component [23]. A simplest form of threshold-based maintenance policies for multi-component systems can be derived from the ones designed for single-component systems, which means that a component is correctively replaced if it is failed or is preventively replaced if its degradation state exceeds a predefined preventive replacement constant. A slightly different version of this policy is employed in [28] for a production system taking into account imperfect preventive maintenance. It should be noted that these maintenance policies are applicable in the case where component dependencies can be neglected. However, they seems not so feasible in practice due to the fact that there exits multiple kinds of dependencies between components in modern systems as analyzed in Section 2.2.

In the literature, several threshold-based maintenance policies have been developed for multi-component systems taking component dependencies into account. Among the component dependencies, the positive economic dependence is usually taken into consideration to take advantage of opportunistic replacement [23]. The most common approach is to consider a preventive and opportunistic threshold for each component [72, 80, 83]. The idea is that a preventive replacement is performed on a component as soon as its degradation approaches its preventive replacement threshold which is independent of the health condition states of the other components. A component can be also replaced along with the replacement of other components in the case where its deterioration reaches its opportunistic threshold which is often lower in comparison to its preventive one. This kind of opportunistic maintenance policy is also used in [20] for a two-component system subjected to stochastic and economic dependence. Clustering replacement opportunities based on different setup-cost levels is considered in [52]. For each level, a threshold is included. A multi-level opportunistic predictive maintenance approach considering both economic and technical structural dependence whereby maintenance of a component requires disassembly of other components is proposed in [48]. Particularly, two opportunistic thresholds are introduced. The objective of the first threshold is to opportunistically select non-disassembled components for corrective/preventive main-

tenance. The second threshold aims to select one or several disassembled components to be opportunistically maintained. A bi-level threshold-based CBM policy utilizing prognostic information for multi-component systems considering stochastic and economic dependence is proposed in [24]. Specifically, the objective of the decision rule at system level is to address whether maintenance actions are needed by considering the system's future reliability based on prognostic results. At component level, the aim is to identify optimal groups of components to be preventively maintained when maintenance is triggered due to the decision made at system level.

It should be noted that the search for optimal threshold-based maintenance policies for multi-component systems is usually performed by Monte-Carlo simulation. If the system is small meaning that there is a few decision variables needed to be optimized, an exhaustive search on fine grid can be used. However, when the system is large, an exhaustive search over the space of all possible threshold values is infeasible. In such cases, heuristic search methods based on genetic or ant algorithms, and particle swarm optimization can be employed to optimize maintenance thresholds [24, 48]. It is important to note that such heuristic algorithms do not guarantee the finding of global optimal solutions, hence it is necessary to run these algorithms several times to get the best results.

It can be also noticed that describing maintenance relations between components via thresholds becomes extremely complicated once maintained systems have complex structure and there exits multiple dependencies between components. This process is often done based on expert knowledge on the number of thresholds to be considered for each components, and on how these thresholds are used to take component dependencies into consideration to be able to obtain optimal maintenance policies. As a result, even if optimization algorithms can search for optimal thresholds, the resulting maintenance policies are not guaranteed to be globally optimal.

2.3.2 Direct-mapping maintenance policies

Within the context of CBM, a decision is made based on new evidence collected from the components of the system under consideration. Therefore, it is natural to think of finding a policy that maps directly from health condition measurements to optimal maintenance actions. Indeed, this kind of policy can be obtained by formulating maintenance decision-making problems as MDPs or other variants which involve the interaction between one or multiple agents (maintenance decision-makers) and a stochastic environment (a model represents the components' degradation process as well as their dependencies in terms of economics, stochasticity and structure through its reward generation and state transition dynamics).

Particularly, maintenance decision-making problem of a single-unit system is formulated as an MDP in [49, 50] under the objective of determining whether to replace the system or not at each discrete decision step. MDPs are also studied for CBM of multi-component systems with different structures in the literature [18, 29, 84, 85]. It should be noted that MDPs' optimal solutions can be found by dynamic programming approaches (e.g., policy and value iteration algorithm) if transition probability matrices are known. However, optimization time becomes very problematic for dynamic programming when the state and action space of a MDP get larger as in the case of multi-component systems [85]. Moreover, MDPs are not always suitable for decision-making in engineering systems due to the requirement of complete information and error-free observations of the system under consideration at every decision epoch [18]. Partially observable MDPs (POMDPs) extends the MDP model by considering partial information during making decisions which in the context of maintenance is caused by the imperfection of inspections [18, 42]. Searching an optimal policy for a POMDP is usually harder than in the standard MDP due to the partial observability and hence is often done by approximate methods such as point-based algorithms as in [25, 43].

It can be noticed that conventional algorithms are just suitable for solving small MDPs or POMDPs due to the curse of dimensionality. Fortunately, recent advancements in the field of deep reinforcement learning (DRL) provide new tools to deal with maintenance decision-making problem of large-scale systems [86, 87]. In particular, double deep Q-network

(DDQN) algorithm is employed in [3] to optimize preventive maintenance policies of multi-component systems considering economic dependence between components and competing failure risks, assuming that several components are physically integrated and have to be maintained simultaneously to reduce the complexity of maintenance decision space. Preventive replacement policy of a serial production line is also optimized by DDQN algorithm in [11]. The state space of this maintenance decision-making problem is extremely large, however, the number of actions is quite small. The authors of [88] employed separate neural networks for estimating each action's value following Q-learning fashion in maintenance operation optimization of power grid systems showing possibilities of reaching global solution. However, this method is only feasible for decision-making applications with relatively small action space size. Proximal policy optimization (PPO) algorithm is combined with imitation learning in [89] to optimize maintenance decisions of a wind farm which are performed by a maintenance crew. A decision taken at any timestep is about the next destination of the maintenance crew (which component to maintain next) implying that the size of action space is equal to the number of components within the system.

Despite the effectiveness of single-agent DRL algorithms in optimizing polices for decision-making problems with large state spaces as shown in the above mentioned works, it is pointed out in the literature that this kind of algorithms is not suitable for maintenance scheduling applications with large action spaces. Fortunately, multi-agent DRL (MADRL) framework can be considered as a promising solution to this issue, which has recently gained great attention in maintenance planning optimization. Particularly, a customized version of DQN algorithm for multi-agent setting is proposed in [90] to optimize maintenance decisions of bridge systems showing the ability to obtain optimal maintenance policies. Maintenance actions of large-scale structures are optimized by deep centralized multi-agent actor-critic (DCMAC) algorithm in [18] considering imperfect maintenance. Agents of the MADRL algorithms mentioned above make decisions individually based on system states/believes which might not be applicable for complex systems where system-wise data collection or transmission might be inefficient or even not feasible during execution [17]. To address this issue, Andriotis et al. [19] formulated maintenance planning tasks as a decentralized partially observable Markov

decision processes (Dec-POMDPs) and modified DCMAC to be a deep decentralized multi-agent actor-critic (DDMAC) algorithm which allow to optimize decentralized maintenance policies in the sense that maintenance decisions of a component or subsystem is made based on its own local history without having to access information at system level. DDMAC is also employed in [91] to optimize inspection and maintenance decisions of multi-component systems whose maintenance decision-making processes are modeled by factored POMDPs which are a special case of the Dec-POMDP model. Dec-POMDP formulation is also considered in [17] for maintenance decision-making problem of large-scale manufacturing systems in which value-decomposition actor-critic (VDAC) algorithm [92] is used to obtain individual preventive maintenance policies.

Table 2.1: Criteria for selecting RL algorithms for maintenance planning optimization of multi-component systems.

	Large state space	Large action space
DRL	✓	✗
MADRL	✓	✓

Table 2.1 gives the criteria to select suitable reinforcement learning algorithms for maintenance planning optimization of multi-component systems. A summary of applications employing DRL and MADRL algorithms for maintenance planing of multi-component systems is presented in Table 2.2.

Table 2.2: Summary of applications using DRL and MADRL for maintenance optimization.

Reference	Application	Algorithm	DRL	MADRL
[3]	Preventive maintenance planning considering economic dependence and competing failure risks	DDQN	✓	

Table 2.2: Summary of applications using DRL and MADRL for maintenance optimization.

Reference	Application	Algorithm	DRL	MADRL
[11]	Maintenance planning for serial production lines with intermediate buffers considering preventive replacement	DDQN	✓	
[88]	Maintenance operation optimization of power grid systems	Adapted DQN	✓	
[89]	Maintenance decision optimization of a wind farm	PPO	✓	
[93]	Distributed flow shop scheduling with flexible maintenance	DQN	✓	
[94]	Maintenance planning for repairable systems subject to degradation and random shock	DQN	✓	
[95]	Rail renewal and maintenance planning considering time, resource resource, and related engineering constraints constraints	DDQN	✓	
[96]	Opportunistic maintenance for multi-unit series systems using proportional hazards models	Modified DQN	✓	
[90]	Maintenance decision optimization for bridge systems	Customized DQN		✓
[18]	Maintenance optimization for large-scale structures considering imperfect maintenance actions	DCMAC		✓
[19]	Inspection and maintenance optimization for deteriorating systems considering life-cycle risk-based constraints and budget limitations	DDMAC		✓

Table 2.2: Summary of applications using DRL and MADRL for maintenance optimization.

Reference	Application	Algorithm	DRL	MADRL
[91]	Inspection and maintenance optimization for deteriorating systems with probabilistic dependencies through Bayesian networks	DDMAC		✓
[17]	Maintenance planning for serial production lines with intermediate buffers considering imperfect preventive maintenance	VDAC		✓

Based on these analysis above, it can be seen that MADRL is a promising framework for optimizing sequential maintenance decisions for multi-component systems with both large state and action spaces. Moreover, it also allows to flexibly adapt maintenance policies to different observability settings (a maintenance decision performed on a component can be made based on information at either component or system level).

2.4 Conclusions

This chapter presented the literature review related to the identified research questions in maintenance planning for systems consisting multiple components subjected to different kinds of dependencies. Based on the developed the-state-of-the-art, the framework of MADRL is identified as a promising solution for maintenance planning optimization of multi-component systems since it allows to deal with the curse of dimensionality and to adapt maintenance policies to different observability settings. However, applying MADRL algorithms for maintenance planning of multi-component systems faces several challenges related to the computation of maintenance costs at system level that are used in the reward generating process without using specific cost values at component level, and the modeling of degradation interactions for discrete-state components that are used to describe the transition dynamic.

These problems lead to two contributions proposed in this PhD as listed below:

1. *Proposal of a maintenance optimization approach based on the framework of MADRL that allows to get rid of the demand of accessing individual costs at component level in the computation of maintenance costs at system level.*

2. *Proposal of a novel model for describing degradation interactions in multi-component systems based on Markov processes.*

Contribution 1 and 2 are respectively presented in Chapter 4 and 5. The next chapter is devoted to give the fundamental concepts of decision-making under uncertainty with the focus on MADRL framework under the objective of facilitating the presentation of the two contributions.

Chapter 3

Multi-agent deep reinforcement learning

3.1 Introduction

This chapter aims at presenting the core concepts of sequential decision making under uncertainty which involves the interaction between one or multiple agents and a stochastic environment.

Single-agent decision-making frameworks are first introduced in Section 3.2. Particularly, the formalization and some basic properties of MDPs are presented in Subsection 3.2.1. Dynamic programming and reinforcement learning approaches that can be used to obtain optimal policies for a MDP are also presented in this subsection. Partially observable MDPs (POMDPs) extend the MDP model by incorporating state uncertainty is formally defined in Subsection 3.2.2

Section 3.3 is devoted to the presentation of multi-agent decision-making frameworks in which the agents cooperate with each other to achieve a common goal. Particularly, decentralized POMDPs (Dec-POMDPs) are introduced in Subsection 3.3.1 which can be used to mathematically describe the interaction between a set of agents and a stochastic environment taking into account both state uncertainty and the effects of uncertainty caused by the agents that must take their decisions in a decentralized way. However, there exists some cases where learning agents can observe ground-truth states of the environment based on which they make their own decisions. In such situations, Dec-POMDPs can be reduced to multi-agent MDPs (MMDPs) which is presented in Subsection 3.3.2.

Finally, Weighted QMIX (WQMIX) algorithm [26] which is originally designed to solve Dec-POMDPs, and a customized version of this algorithm constructed to effectively solve MMDPs are presented in Subsection 3.3.3

3.2 Single-agent decision-making frameworks

3.2.1 Markov decision processes

A MDP can be used to mathematically model a sequential decision-making process of an agent (decision-maker) in an uncertain environment as illustrated in Figure 3.1. Specifically, at timestep t_k, the agent observes a true state of the environment $\boldsymbol{s}_k \in \mathcal{S}$. Based on the observed state, the agent chooses an action $\boldsymbol{a}_k \in \mathcal{A}$ to execute. Once the chosen action is executed, the environment stochastically transitions to a new state $\boldsymbol{s}_{k+1} \in \mathcal{S}$ at timestep t_{k+1} and a reward $r_k \in \mathbb{R}$ is emitted to the agent. The formal definition of a MDP is given as belows:

Definition 3.1 (MDP)**.** A Markov decision process is defined by the tuple $\{\mathcal{S}, \mathcal{A}, p^s, r\}$ where:

- $\mathcal{S}$ is the state space that is a set of all possible states of the environment;
- $\mathcal{A}$ is the action space that is a set of all possible actions from which the agent can choose to interact with the environment;
- $p^s(\boldsymbol{s}_k, \boldsymbol{a}_k, \boldsymbol{s}_{k+1}) = \Pr(\boldsymbol{S}_{k+1} = \boldsymbol{s}_{k+1} \mid \boldsymbol{S}_k = \boldsymbol{s}_k, \boldsymbol{A}_k = \boldsymbol{a}_k)$ is the transition probability of reaching state $\boldsymbol{s}_{k+1} \in \mathcal{S}$ at timestep t_{k+1} after executing action $\boldsymbol{a}_k \in \mathcal{A}$ in state $\boldsymbol{s}_k \in \mathcal{S}$ at timestep t_k. This function is called the system dynamic in the literature;
- $r(\boldsymbol{s}_k, \boldsymbol{a}_k, \boldsymbol{s}_{k+1})$ is the reward function that generates a scalar reward $r_k \in \mathbb{R}$ at timestep t_k for the agent when taking action $\boldsymbol{a}_k \in \mathcal{A}$ in state $\boldsymbol{s}_k \in \mathcal{S}$ and the environment then reaches state $\boldsymbol{s}_{k+1} \in \mathcal{S}$ at next timestep.

Given a MDP, the objective is to find decision rules for each timestep that optimize predefined objectives, e.g., reaching some goal states, maximizing the expected cumulative reward within a finite or infinite planning horizon. In this thesis, we concentrate on finding a policy that maximizes the *expected accumulated discounted rewards over an infinite horizon*. In the context of maintenance planning, this kind of policy can be considered as an optimal long-

term maintenance plan.

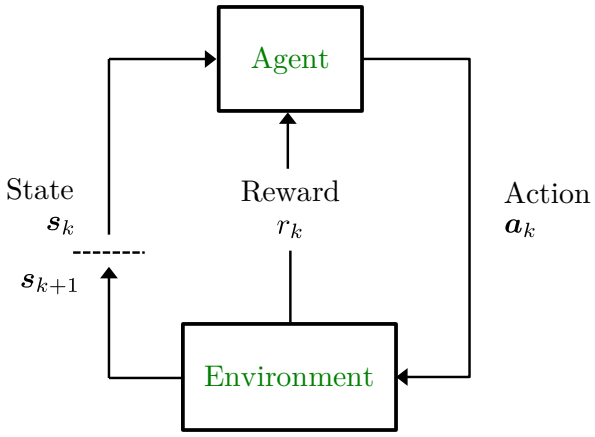


Figure 3.1: Illustration of a MDP.

3.2.1.1 Bellman equations

A policy $\boldsymbol{\pi}(\boldsymbol{a}_k|\boldsymbol{s}_k) = \Pr(\boldsymbol{A}_k = \boldsymbol{a}_k \mid \boldsymbol{S}_k = \boldsymbol{s}_k)$ is generally a mapping from states to probabilities of selecting all possible actions. Each policy is associated with a corresponding total return $G_k = \sum_{l=0}^{\infty} \gamma^l . r(\boldsymbol{S}_{k+l}, \boldsymbol{A}_{k+l}, \boldsymbol{S}_{k+1+l})$, where $\gamma \in [0, 1)$ is the discount factor used to balance the impact of immediate and future rewards as well as to guarantee that the objective function is bounded, is the total discounted reward collected under this policy, from any timestep t_k to the end of the planning horizon.

The value of a state under a policy $\boldsymbol{\pi}$ is expressed by the *value function* (state-value function) defined as:

$$\begin{aligned}
\boldsymbol{V}(\boldsymbol{s}_k) &= \mathbb{E}\left[G_k \mid \boldsymbol{S}_k = \boldsymbol{s}_k\right] \\
&= \mathbb{E}\left[r(\boldsymbol{S}_k, \boldsymbol{A}_k, \boldsymbol{S}_{k+1}) + \gamma . G_{k+1} \mid \boldsymbol{S}_k = \boldsymbol{s}_k\right] \\
&= \mathbb{E}\left[r(\boldsymbol{S}_k, \boldsymbol{A}_k, \boldsymbol{S}_{k+1}) + \gamma . \boldsymbol{V}_{\pi}(\boldsymbol{S}_{k+1}) \mid \boldsymbol{S}_k = \boldsymbol{s}_k\right] \\
&= \sum_{\boldsymbol{a}_k} \boldsymbol{\pi}(\boldsymbol{a}_k|\boldsymbol{s}_k) \sum_{\boldsymbol{s}_{k+1}} p^s(\boldsymbol{s}_k, \boldsymbol{a}_k, \boldsymbol{s}_{k+1}) . \left[r(\boldsymbol{s}_k, \boldsymbol{a}_k, \boldsymbol{s}_{k+1}) + \gamma . \boldsymbol{V}(\boldsymbol{s}_{k+1})\right]
\end{aligned} \tag{3.1}$$

Similarly, the quality of choosing an action at a specific state following a policy $\boldsymbol{\pi}$ is described through the *action-value function* (state-action value function):

$$
\begin{aligned}
\boldsymbol{Q}(\boldsymbol{s}_k, \boldsymbol{a}_k) &= \mathbb{E}\left[G_k \mid \boldsymbol{S}_k = \boldsymbol{s}_k, \boldsymbol{A}_k = \boldsymbol{a}_k\right] \\
&= \mathbb{E}\left[r(\boldsymbol{S}_k, \boldsymbol{A}_k, \boldsymbol{S}_{k+1}) + \gamma.G_{k+1} \mid \boldsymbol{S}_k = \boldsymbol{s}_k, \boldsymbol{A}_k = \boldsymbol{a}_k\right] \\
&= \mathbb{E}\left[r(\boldsymbol{S}_k, \boldsymbol{A}_k, \boldsymbol{S}_{k+1}) + \gamma.\boldsymbol{V}(\boldsymbol{S}_{k+1}) \mid \boldsymbol{S}_k = \boldsymbol{s}_k, \boldsymbol{A}_k = \boldsymbol{a}_k\right] \\
&= \sum_{\boldsymbol{s}_{k+1}} p^s(\boldsymbol{s}_k, \boldsymbol{a}_k, \boldsymbol{s}_{k+1}).\left[r(\boldsymbol{s}_k, \boldsymbol{a}_k, \boldsymbol{s}_{k+1}) + \gamma.\boldsymbol{V}(\boldsymbol{s}_{k+1})\right] \\
&= \sum_{\boldsymbol{s}_{k+1}} p^s(\boldsymbol{s}_k, \boldsymbol{a}_k, \boldsymbol{s}_{k+1}).\left[r(\boldsymbol{s}_k, \boldsymbol{a}_k, \boldsymbol{s}_{k+1}) + \gamma.\sum_{\boldsymbol{a}_{k+1}} \boldsymbol{\pi}(\boldsymbol{a}_{k+1}|\boldsymbol{s}_{k+1})\boldsymbol{Q}(\boldsymbol{s}_{k+1}, \boldsymbol{a}_{k+1})\right]
\end{aligned}
\tag{3.2}
$$

The recursive forms in equation (3.1) and (3.2) are respectively the Bellman equations for value function and action-value function for a policy $\boldsymbol{\pi}$. Under standard conditions for discounted MDPs, out of all possible policies there exists at least one deterministic policy, $\boldsymbol{\pi}^*$ that is optimal, in other words, that maximizes $\boldsymbol{V}(\boldsymbol{s}_k)$ for all $\boldsymbol{s}_k \in \mathcal{S}$. The Bellman equations for the value and action-value function of a such optimal policy can be expressed in the following equations:

$$
\boldsymbol{V}^*(\boldsymbol{s}_k) = \max_{\boldsymbol{a}_k} \sum_{\boldsymbol{s}_{k+1}} p^s(\boldsymbol{s}_k, \boldsymbol{a}_k, \boldsymbol{s}_{k+1}).\left[r(\boldsymbol{s}_k, \boldsymbol{a}_k, \boldsymbol{s}_{k+1}) + \gamma.\boldsymbol{V}^*(\boldsymbol{s}_{k+1})\right] \tag{3.3}
$$

$$
\boldsymbol{Q}^*(\boldsymbol{s}_k, \boldsymbol{a}_k) = \sum_{\boldsymbol{s}_{k+1}} p^s(\boldsymbol{s}_k, \boldsymbol{a}_k, \boldsymbol{s}_{k+1}).\left[r(\boldsymbol{s}_k, \boldsymbol{a}_k, \boldsymbol{s}_{k+1}) + \gamma.\max_{\boldsymbol{a}_{k+1}} \boldsymbol{Q}^*(\boldsymbol{s}_{k+1}, \boldsymbol{a}_{k+1})\right] \tag{3.4}
$$

3.2.1.2 Dynamic programming

The family of dynamic programming algorithms can be used to obtain an optimal solution for a MDP in the case where the system dynamic, i.e., $p^s(\boldsymbol{s}_k, \boldsymbol{a}_k, \boldsymbol{s}_{k+1})$, is known before hand. Among the algorithms of this family, the value iteration (VI) algorithm which is directly constructed based on equation (3.3) is widely used in practice due to its simplicity. The pseudo code of VI is given in algorithm 1.

It should be noted that VI is capable of finding an exact solution for a MDP if its transition dynamic is known and its size is relatively small. However, VI is not suitable for solving large MDPs because its computational time is very problematic. This problem is shown via simulations conducted in Section 4.4 and 5.4.

Algorithm 1 Value iteration

Initialize a small threshold $\theta > 0$ for determining estimation accuracy

Initialize an array $\boldsymbol{V}(\boldsymbol{s}_k)$ for holding estimated values of all states in the state space

Initialize $\Delta > \theta$

while $\Delta > \theta$ **do**

    for each $\boldsymbol{s}_k$ **do**

        $\boldsymbol{v} \leftarrow \boldsymbol{V}(\boldsymbol{s}_k)$

        $\boldsymbol{V}(\boldsymbol{s}_k) \leftarrow \max_{\boldsymbol{a}_k} \sum_{\boldsymbol{s}_{k+1}} p^s(\boldsymbol{s}_k, \boldsymbol{a}_k, \boldsymbol{s}_{k+1}).\left[r(\boldsymbol{s}_k, \boldsymbol{a}_k, \boldsymbol{s}_{k+1}) + \gamma.\boldsymbol{V}(\boldsymbol{s}_{k+1})\right]$

        $\Delta \leftarrow \max(\Delta, |\boldsymbol{v} - \boldsymbol{V}(\boldsymbol{s}_k)|)$

Optimal policy $\boldsymbol{\pi}^*(\boldsymbol{s}_k) = \operatorname{argmax}_{\boldsymbol{a}_k} \sum_{\boldsymbol{s}_{k+1}} p^s(\boldsymbol{s}_k, \boldsymbol{a}_k, \boldsymbol{s}_{k+1}).\left[r(\boldsymbol{s}_k, \boldsymbol{a}_k, \boldsymbol{s}_{k+1}) + \gamma.\boldsymbol{V}(\boldsymbol{s}_{k+1})\right]$

3.2.1.3 Deep reinforcement learning

If the system dynamic is unknown, one can use reinforcement learning (RL) algorithms to optimize policies for MDPs. In this thesis, we focus on model-free value-based RL algorithms due to their implementation simplicity in practice. Particularly, this class of RL algorithms learns to approximate action-value functions by sequentially interacting with the environment to gain experiences without any prior knowledge about the environment dynamic.

Traditional value-based RL algorithms such as Q-learning [97] and double Q-learning [98] represent the action-value function under a tabular form (i.e., value of each state-action pair takes a slot in the Q-table) and try to update the table via computing temporal difference errors. One drawback of these algorithms is that they cannot be applied for decision-making problems with large state and action spaces because their action-value function's tabular representation require a lot of memories to store Q-values, and due to the requirement of having large enough iterations through all state-action pairs to be able to obtain optimal policies [99].

Recently, a combination of RL and deep learning has created a new filed called deep RL (DRL) which provides a powerful framework to tackle MDPs with large state spaces [18].

The very first success of DRL is in learning to master games [100]. Particularly, Deep Q-network (DQN) algorithm, which uses artificial neural networks (Q-networks) to approximate action-value functions, can learn to play games at human level. It should be noted that the success of DQN is also due to the use of a replay buffer which has fixed length and function following first-in first-out mechanism [101]. Specifically, this kind of memory allows not only to gain the independent and identically distributed assumption required by gradient-descent-based algorithms but also to exploit rare experiences to update deep Q-networks more than once. Additionally, the use of a target Q-network for the update of the main Q-network also helps stabilize the training process [100]. In particular, the main Q-network also known as the online or policy network is used for selecting an action given a current state, and is updated frequently. In contrast, the target network whose weights are periodically copied from the policy network, aims at computing the target for updating the weights of the policy network.

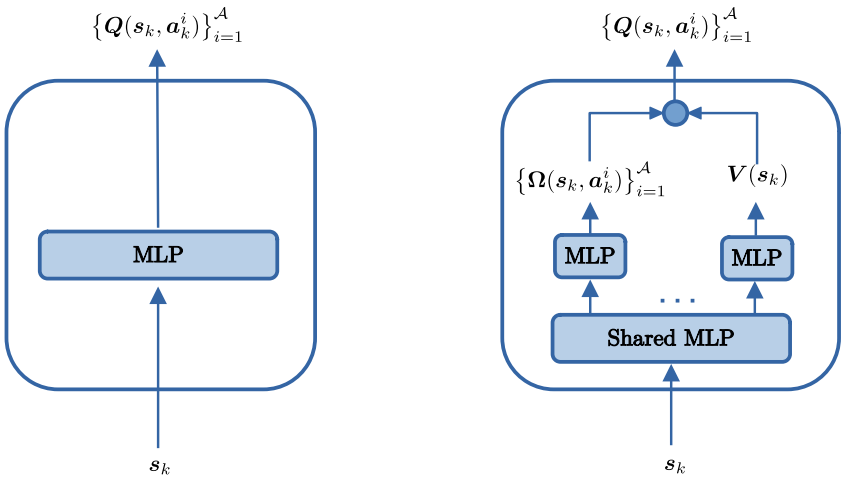


Figure 3.2: (a) Illustration of a traditional single stream Q-network (b) Illustration of a dueling network containing two separate streams to estimate state and advantage values which are then aggregated to compute Q-values.

There are in the literature some variants of DQN designed to improve its performance, how-

ever, the core components of these algorithms are the ANN-based representation of action-value functions, the use of replay buffers and target networks. Particularly, double DQN algorithm constructed based on the theory of double estimators is proposed in [102] to deal with the large overestimation of action-value functions in noisy environments. The dueling network proposed in [103] is composed of two separate estimators for the state-value and advantage function which are combined via a special aggregating layer to produce an estimate of the action-value function. The dueling network is useful in cases when the differences between Q-values for a given state are very small relative to their magnitude. An illustration of a traditional Q-network and a dueling network is given in Figure 3.2.

3.2.2 Partially observable Markov decision processes

MDPs provide ways for modeling sequential decision-making problems in stochastic environments with uncertain action outcomes and exact observations [18]. However, it is difficult for an agent in some cases to observe true states of the environment due to noisy and limited sensors. To address this issue, POMDPs extend the MDP model by incorporating state uncertainty [42, 43].

Particularly, at a decision step t_k, an agent no longer knows the exact state of the environment $\boldsymbol{s}_k \in \mathcal{S}$, instead an observation $\boldsymbol{o}_k \in \mathcal{O}$ which is a partial representation of $\boldsymbol{s}_k$. Therefore, it has to maintain a belief $\boldsymbol{b}_k = \Pr(\boldsymbol{S}_k = \boldsymbol{s}_k \mid \boldsymbol{H}_k = \boldsymbol{h}_k)$, where $\boldsymbol{h}_k = \{\boldsymbol{o}_0, \boldsymbol{a}_0, \ldots, \boldsymbol{a}_{k-1}, \boldsymbol{o}_k\} \in \mathcal{H}$ is the history of the agent which keeps track all past observations and actions taken, about all states of the environment (a probabilistic distribution over all possible states) and use this information to decide upon an action $\boldsymbol{a}_k \in \mathcal{A}$.

An illustration of a POMDP is depicted in Figure 3.3 and its formal definition is given as belows:

Definition 3.2 (POMDP)**.** A partially observable Markov decision process is defined by the tuple $\{\mathcal{S}, \mathcal{A}, \mathcal{O}, p^s, p^o, r\}$ in which $\mathcal{S}$, $\mathcal{A}$, p^s, r have the same definition as in MDPs and:

- $\mathcal{O}$ is the observation space that is a set of all possible observations an agent can receive

from the environment;

- $p^o(\boldsymbol{o}_k, \boldsymbol{s}_k) = \Pr(\boldsymbol{O}_k = \boldsymbol{o}_k \mid \boldsymbol{S}_k = \boldsymbol{s}_k)$ is the probability of receiving observation $\boldsymbol{o}_k \in \mathcal{O}$ from state $\boldsymbol{s}_k \in \mathcal{S}$ at timestep t_k.

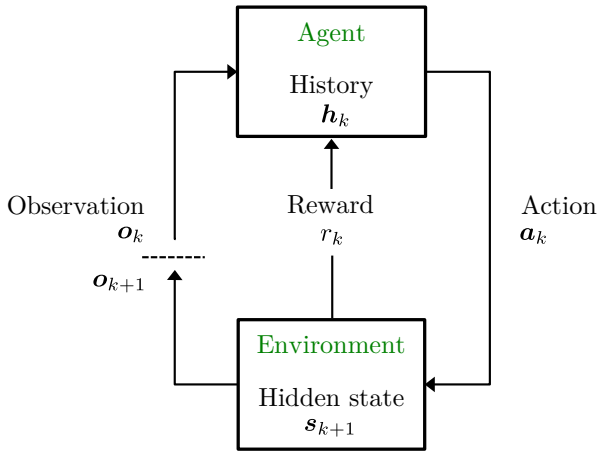


Figure 3.3: Illustration of a POMDP.

If the transition models are known, the belief at timestep t_k can be computed efficiently by using the belief and action taken at last timestep, and the recent observation received based on the Markov property and the Bayesian update [18]. Without the transition models, an approximation of belief (agent's internal state) can be computed via sampling by using recurrent neural networks, which allows to derive decision rules [104].

3.3 Multi-agent decision-making frameworks

3.3.1 Decentralized partially observable Markov decision processes

The Dec-POMDP framework extends the POMDP model to decision-making problems of multi-agent systems. In particular, Dec-POMDPs allow to mathematically describe the interaction between multiple agents and a stochastic environment taking into consideration both state uncertainty and the effects of uncertainty caused by the agents that must take their decisions in a decentralized way, i.e., based on their own individual observations [104].

An illustration of the interaction between three agents and their environment is given in Figure 3.4. A Dec-POMDP is formally defined as follows:

Definition 3.3 (Dec-POMDP)**.** A decentralized partially observable Markov decision process is defined by the tuple $\{\mathcal{AG}, \mathcal{S}, \mathcal{A}, \mathcal{O}, p^s, p^o, r\}$ where:

- $\mathcal{AG} = \left\{AG^1, AG^2, \ldots, AG^N\right\}$ is the set of N agents interacting with the environment;
- $\mathcal{S}$ is the state space that is a set of all possible states of the environment;
- $\mathcal{A} = \mathcal{A}^1 \times \mathcal{A}^2 \times ... \times \mathcal{A}^N$ is the joint action space constituted by the agents' individual action space $\mathcal{A}^i$ $(i = 1, \ldots, N)$;
- $\mathcal{O} = \mathcal{O}^1 \times \mathcal{O}^2 \times ... \times \mathcal{O}^N$ is the joint observation space that is a set of all possible joint observations received from the environment, and is constituted by the agents' individual observation space $\mathcal{O}^i$ $(i = 1, \ldots, N)$;
- $p^s(\boldsymbol{s}_k, \boldsymbol{a}_k, \boldsymbol{s}_{k+1}) = \Pr(\boldsymbol{S}_{k+1} = \boldsymbol{s}_{k+1} \mid \boldsymbol{S}_k = \boldsymbol{s}_k, \boldsymbol{A}_k = \boldsymbol{a}_k)$ is the transition probability of reaching state $\boldsymbol{s}_{k+1} \in \mathcal{S}$ at timestep t_{k+1} by executing a joint action $\boldsymbol{a}_k \in \mathcal{A}$ in state $\boldsymbol{s}_k \in \mathcal{S}$ at timestep t_k;
- $p^o(\boldsymbol{o}_k, \boldsymbol{s}_k) = \Pr(\boldsymbol{O}_k = \boldsymbol{o}_k \mid \boldsymbol{S}_k = \boldsymbol{s}_k)$ is the probability of receiving joint observation $\boldsymbol{o}_k \in \mathcal{O}$ from state $\boldsymbol{s}_k \in \mathcal{S}$ at timestep t_k;
- $r(\boldsymbol{s}_k, \boldsymbol{a}_k, \boldsymbol{s}_{k+1})$ is the reward function that generates rewards for the agents when executing joint action $\boldsymbol{a}_k$ in state $\boldsymbol{s}_k$ at timestep t_k and the environment then reaches state $\boldsymbol{s}_{k+1}$ at next timestep. The reward r_k at any timestep t_k is shared by all agents (cooperative multi-agent setting).

At decision time t_k, each agent $AG^i \in \mathcal{AG}$ receives an individual observation $o_k^i \in \mathcal{O}^i$ from the environment which cannot be shared with each other. The agents separately choose individual actions $(a_k^i \in \mathcal{A}^i)$ based on their individual policy (π^i) and the information they have stored in individual histories $(h_k^i \in \mathcal{H}_k^i)$. The individual actions form a joint action $\boldsymbol{a}_k \in \mathcal{A}$ which makes the environment transition to a new state $\boldsymbol{s}_{k+1}$ at next timestep t_{k+1}.

Solving a Dec-POMDP (decentralized training and decentralized execution) is not an easy

task due the uncertainty of state observability and the uncertainty caused by the interaction between individual agents. However, in some cases where the training phase is not necessarily to be as strict as the execution phase, one can take advantage of additional formation available during the training. In such cases, the Centralized Training for Decentralized Execution (CTDE) paradigm can be employed, which has been recently received great attention in the field of cooperative MADRL. Among the algorithms that employ this paradigm, Weighted QMIX (WQMIX) is a model-free value-based MADRL algorithm which shows good performance on many standard benchmarks [26]. Therefore, it is used in this PhD to optimize maintenance decisions for multi-component systems. The detail presentation of WQMIX is given in Section 3.3.3.

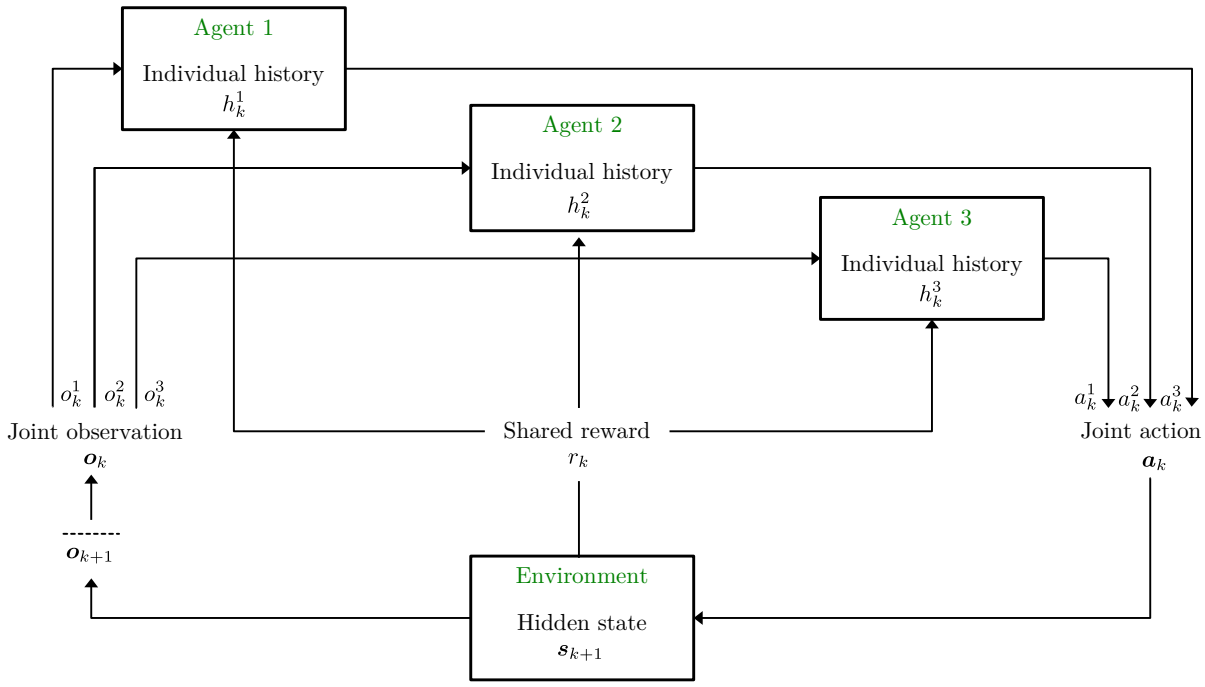


Figure 3.4: Illustration of a Dec-POMDP modeling the interaction between three agents and a stochastic environment.

3.3.2 Multi-agent Markov decision processes

Dec-POMDPs provide a general way to model the interaction between multiple agents and an uncertain environment by considering both joint partial observations of the ground-truth system state and joint actions. However, in some cases where agents can observe ground-truth states of the environment based on which they make their own decisions. Such global information can arise due to either full observability or communications [104]. In such situations, a Dec-POMDP can be simplified to a multi-agent MDP (MMDP) [105] whose formal definition is given as follows:

Definition 3.4 (MMDP)**.** A multi-agent Markov decision process is defined by the tuple $\{\mathcal{AG}, \mathcal{S}, \mathcal{A}, p^s, r\}$ in which $\mathcal{S}$, p^s, r have the same definition as in MDPs, and $\mathcal{AG}$, $\mathcal{A}$ have the same definition as in Dec-POMDPs.

It should be noted that a MMDP can be viewed as a regular MDP with a single agent that selects joint actions. Therefore, it can be solved by dynamic programming algorithms if the underlying transition probabilities are known or by reinforcement learning algorithms otherwise. However, this single-agent view is only applicable for MMDPs with small number of agents due to the fact that the action space's size of a MMDP grows exponentially in the number of agents.

3.3.3 WQMIX algorithm

This section is devoted to the presentation of WQMIX algorithm [26] which can be used to obtain optimal policies for Dec-POMDPs and MMDPs. In this PhD, Dec-POMDPs and MMDPs are employed to model multi-agent maintenance decision-making tasks in which agents can partially and fully observe the joint state of the environment respectively (partially and fully observable setting). Therefore, we first present the main idea and core components of this MADRL algorithm for the former scenario in Subsection 3.3.3.1. A customized version of WQMIX for the later scenario is then depicted in Subsection 3.3.3.2.

3.3.3.1 Partially observable setting

Two main components of WQMIX algorithm are the monotonic factorization scheme of the joint action-value function and the weighting scheme in the loss function used to update learning agents' Q-network. The factorization scheme aims at addressing the decision selection consistency in the sense that cooperative learning agents try to choose a common decision $\boldsymbol{a}_k$ from their joint action space according to the joint action-value function that should be consistent with the combination of individual actions chosen by each agent separately from its own action space, which is mathematically expressed as:

$$\underset{\boldsymbol{a}_k \in \mathcal{A}}{\operatorname{argmax}}\, \boldsymbol{Q}^{tot}(\boldsymbol{h}_k, \boldsymbol{a}_k) = \begin{bmatrix} \underset{a_k^1 \in \mathcal{A}^1}{\operatorname{argmax}}\, Q^1(h_k^1, a_k^1) & \underset{a_k^2 \in \mathcal{A}^2}{\operatorname{argmax}}\, Q^2(h_k^2, a_k^2) & \cdots & \underset{a_k^N \in \mathcal{A}^N}{\operatorname{argmax}}\, Q^N(h_k^N, a_k^N) \end{bmatrix} \tag{3.5}$$

in which $\boldsymbol{Q}^{tot}(\boldsymbol{h}_k, \boldsymbol{a}_k)$ is an approximate of $\boldsymbol{Q}(\boldsymbol{s}_k, \boldsymbol{a}_k)$. This factorization method assumes that the joint action-value function can be decomposed into the per-agent ones and is a continuous monotonic function of them [106]:

$$\frac{\partial \boldsymbol{Q}^{tot}(\boldsymbol{h}_k, \boldsymbol{a}_k)}{\partial Q^i(h_k^i, a_k^i)} \geq 0, \forall i \in \{1, ..., N\} \tag{3.6}$$

This assumption can be realized in practice by employing a mixing network which is a multilayer perceptron (MLP) taking individual agents' own Q-values as input and outputting values of the approximate system action-value function $\boldsymbol{Q}^{tot}(\boldsymbol{h}_k, \boldsymbol{a}_k)$, whose weights are generated by a hyper network [107] to guarantee their values to be positive.

However, the monotonic factorization scheme restricts an agent choosing its own action independent of the actions chosen by other agents. This may lead to the finding of suboptimal solutions for decision-making applications requiring strong cooperation effort between learning agents. The authors of WQMIX in [26] showed that this limit emanates from the equal weighting placed on each join action in the loss function that is used to update the joint Q-function, and then proposed a special weighting method to address this issue. In particular, a feedforward network without any restriction to its weights is used to estimate $\boldsymbol{Q}(\boldsymbol{s}_k, \boldsymbol{a}_k)$ values which are then employed as baselines to put more attention on potential optimal joint actions in the loss function. The losses of one transition sample, $(\boldsymbol{s}_k, \boldsymbol{a}_k, r_k, \boldsymbol{s}_{k+1})$, used to

update the networks representing $\boldsymbol{Q}^{tot}(\boldsymbol{h}_k, \boldsymbol{a}_k)$ and $\boldsymbol{Q}(\boldsymbol{s}_k, \boldsymbol{a}_k)$ are computed as belows:

$$\begin{aligned} L^{\boldsymbol{Q}^{tot}} &= w(\boldsymbol{s}_k, \boldsymbol{a}_k)(\boldsymbol{Q}^{tot}(\boldsymbol{h}_k, \boldsymbol{a}_k) - y)^2 \\ L^{\boldsymbol{Q}} &= (\boldsymbol{Q}(\boldsymbol{s}_k, \boldsymbol{a}_k) - y)^2 \end{aligned} \tag{3.7}$$

where:

- $y = r + \gamma \boldsymbol{Q}_{target}(\boldsymbol{s}_k, \text{argmax}_{\boldsymbol{a}_{k+1}} \boldsymbol{Q}^{tot}_{target}(\boldsymbol{h}_{k+1}, \boldsymbol{a}_{k+1}))$ is the fixed target with $\boldsymbol{Q}_{target}$ and $\boldsymbol{Q}^{tot}_{target}$ are computed from the target networks of $\boldsymbol{Q}$ and $\boldsymbol{Q}^{tot}$ respectively.
- $w(\boldsymbol{s}_k, \boldsymbol{a}_k)$ is the weight of the joint action $\boldsymbol{a}_k$ whose value is equal to 1 if $\boldsymbol{Q}^{tot}(\boldsymbol{h}_k, \boldsymbol{a}_k) < y$ or equal to $\alpha \in (0, 1]$. This weighting mechanism is called "optimistic weighting" because it assigns higher weighting to an action, $\boldsymbol{a}_k$, which might be optimal if its corresponding $\boldsymbol{Q}^{tot}(\boldsymbol{s}_k, \boldsymbol{a}_k)$ is underestimated relative to the true value that is replaced by y [26].

The general architecture of WQMIX is illustrated in Figure 3.5 and the structure of local agent networks is depicted in Figure 3.6 which is incorporated with a recurrent neural network (e.g. Gated Recurrent Unit) aiming at memorizing the history of local state-action transitions. Moreover, it should be noted that the CTDE paradigm is used for training WQMIX agents in this scenario which means that learning agents are assumed to be able to access the global state during training, however, they make decisions based on its own local histories in execution phase.

3.3.3.2 Fully observable setting

In this scenario, local agents are capable of accessing the global state during both training and execution phase. Therefore, we propose to replace separate agent networks used in the partially observable setting by a single branching dueling network (branching network for short) [108] to take advantage of the common joint state. The special structure of the branching network allows to achieve a linear increase in the size of deep Q-networks' output layer to avoid the curse of dimensionality. Furthermore, it also allows creating virtual communication channels between cooperative learning agents to facilitate decision-making processes as well as to avoid the use of recurrent neural networks to memorize local transition histories in

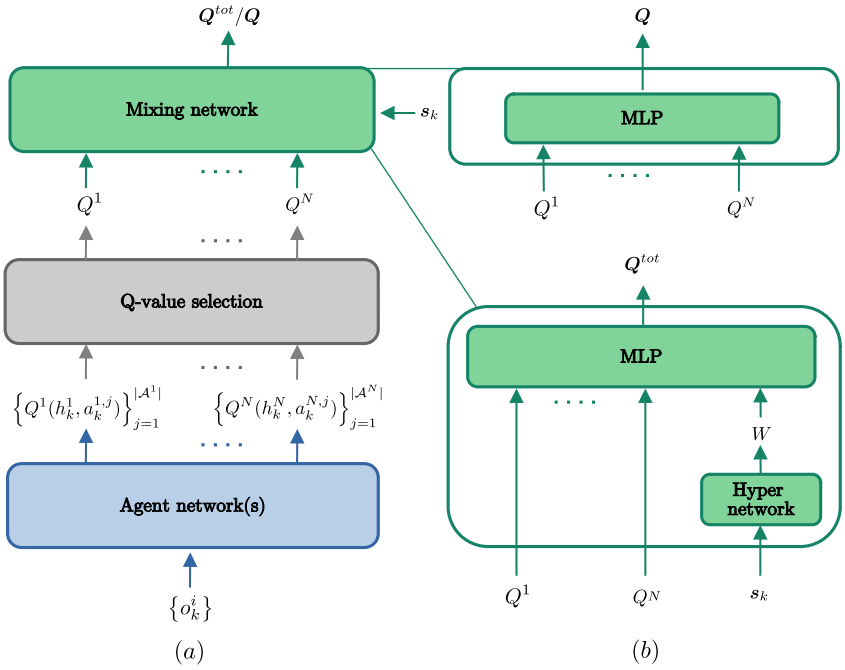


Figure 3.5: (a) Illustration of WQMIX architecture. Each agent $AG^i \in \mathcal{AG}$ computes Q-values for each local action, $\left\{Q^i(h_k^i, a_k^{i,j})\right\}_{j=1}^{|\mathcal{A}^i|}$, based on the local input observation. After that, the maximal Q-value of each agent is selected. The combination of those maximal Q-values is then fed to a mixing network to compute $\boldsymbol{Q}^{tot}$ or $\boldsymbol{Q}$. (b) Illustration of mixing network structures. The weights of the mixing network used for computing $\boldsymbol{Q}^{tot}$ are generated by a hyper-network to guarantee their values to be positive, whereas the one used for calculating $\boldsymbol{Q}$ is a MLP without any restriction to its weights.

the partially observable setting that may slow down the learning process as shown through experiments conducted in Section 5.4.

The structure of the branching network is depicted in Figure 3.6b. It can be noticed that the system state is considered as input for all agents, however, outputs of each branch of the branching network are still individual action values of corresponding agents. Specially, the joint state, $\boldsymbol{s}_k$, is first passed through the shared MLP to encode latent representations between learning agents that is then used to calculate the common joint state value, $\boldsymbol{V}(\boldsymbol{s}_k)$, as well as the individual advantage values, $\Omega^i(\boldsymbol{s}_k, a_k^i)$ $(i \in \{1, ..., N\})$. After that, the agents'

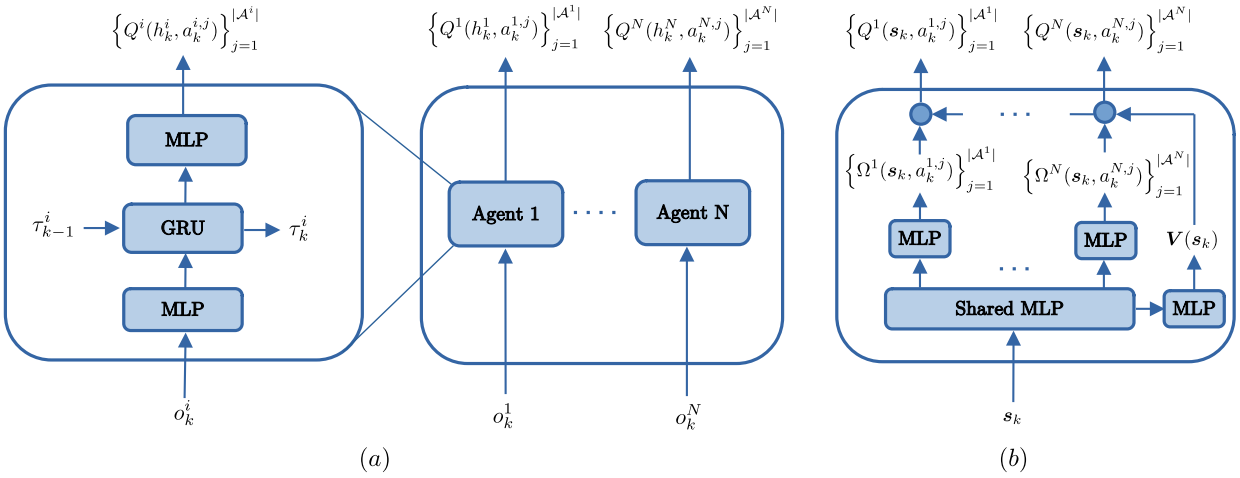


Figure 3.6: (a) Illustration of the structure of separate agent networks used in partially observable setting. (b) Illustration of the structure of a single branching network used in fully observable setting.

local action values are computed by aggregating the joint state value and the advantage values in each branch by using the following equation [108]:

$$Q^i(\boldsymbol{s}_k, a_k^i) = \boldsymbol{V}(\boldsymbol{s}_k) + \Omega^i(\boldsymbol{s}_k, a_k^i) - \frac{1}{|\mathcal{A}^i|} \sum_{a_k \in \mathcal{A}^i} \Omega^i(\boldsymbol{s}_k, a_k^i) \tag{3.8}$$

It is worthy noting that since the gradient of all branches is backpropagated through the shared MLP, the combined gradient prior to entering the deepest layer of this network is re-scaled in practice by a factor of $1/\sqrt{N+1}$ to improve the stability of training processes. Moreover, it should be noted that the general architecture of WQMIX and the computational process of $\boldsymbol{Q}^{tot}$ and $\boldsymbol{Q}$ are the same for both partially and fully observable context.

3.4 Conclusions

In this chapter, MDPs and POMDPs are first introduced to mathematically describe the sequential decision making process of an agent inhabiting in a stochastic environment where the agent can fully and partially observe states of the environment respectively. Dynamic programming and reinforcement learning approaches are also presented which can be used

to find an optimal policie for a MDP that defines how an agent should act to accumulate as much reward as possible over time. After that, the concept of multi-agent decision-making is presented through the formalization of Dec-POMDPs and MMDPs. Finally, the core components of WQMIX algorithm [26], which can be used to solve Dec-POMDPs and MMDPs, are presented.

Chapter 4

Proposal of an AI-based maintenance planning approach

4.1 Introduction

In this chapter, an AI-based maintenance planning approach based on the framework of MADRL is proposed for multi-component systems. The proposed approach allows to effectively optimize maintenance decisions for systems consisting of multiple components as well as to take into consideration the first scientific issue related to the requirement of individual cost models at component level to construct maintenance cost models at system level.

Particularly, the maintenance cost model at system level is first learned by an ANN from a dedicated condition monitoring dataset. The learned ANN-based maintenance cost model along with the degradation model of the system under consideration are then used to construct an interactive environment for cooperative reinforcement learning agents. Finally, MADRL algorithms are employed to optimize maintenance decisions by letting cooperative agents to interact with the constructed environment.

The chapter is organized as follows. Section 4.2 is devoted to the description of the system studied in this chapter. The proposed AI-based approach for maintenance decision optimization of multi-component systems is presented in Section 4.3. Numerical simulations are conducted on Section 4.4 to investigate the performance as well as the scalability of the proposed maintenance approach. Conclusions and some perspectives are presented in the last section.

4.2 System description

4.2.1 System degradation

We consider a general system containing N components which are inter-connected according to a specific structure (e.g., series, parallels or mixed) from a reliability block diagram point of view. The system is monitored periodically at time $t_k = k \times \Delta_t$ $(k = 0, 1, 2, ...)$ where Δ_t is the inter-inspection time. In addition, component i $(i = 1, \ldots, N)$ has $(m^i + 1)$ discrete health condition states including a new state, $(m^i - 1)$ degraded states and a failed state. By denoting s_k^i is the state of component i at inspection time t_k, the component's state can be expressed as:

$$s_k^i = \begin{cases} 0 & \text{if component } i \text{ is as good as new} \\ j & \text{if component } i \text{ is in degraded state } j \ (1 \leq j \leq m^i - 1) \\ m^i & \text{if component } i \text{ fails} \end{cases} \tag{4.1}$$

It should be noted that the higher j, the more degraded the component is. Moreover, the state transition (degradation) of component i $(i = 1, \ldots, N)$ in the absence of maintenance intervention between two consecutive inspections is assumed to follow a Markov process whose form is given as bellows:

$$\tilde{\boldsymbol{P}}^i = \begin{bmatrix} \tilde{p}_{00}^i & \tilde{p}_{01}^i & \tilde{p}_{02}^i & \cdots & \tilde{p}_{0m^i}^i \\ 0 & \tilde{p}_{11}^i & \tilde{p}_{12}^i & \cdots & \tilde{p}_{1m^i}^i \\ 0 & 0 & \tilde{p}_{22}^i & \cdots & \tilde{p}_{2m^i}^i \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix} \tag{4.2}$$

in which $\tilde{p}_{uv}^i$ is a non-negative real number representing the degradation transition probability of component i from state u to v that satisfies: $\sum_{v=u}^{m^i} \tilde{p}_{uv}^i = 1, \forall u \in \{0, 1, \ldots, m^i\}$. Moreover, it is also supposed that the components' degradation process are independent.

4.2.2 Maintenance operations

Both corrective and preventive maintenance strategy are considered for the studied system. Particularly, a corrective maintenance intervention is carried out to replace a failed component, whereas the preventive one is performed on a functioning component to avoid its failure as well as the system's failure that could be either perfect or imperfect. A perfect preventive maintenance action completely restores a survival component to be "as good as new". Conversely, the imperfect one implies that component's state after maintenance is somewhere between its state before maintenance and "as good as new" state. Moreover, it is important to note that maintenance interventions can only be discretely carried out after components' health condition states are revealed by inspections. As a result, if the failure of a component occurs between two successive inspections, corrective maintenance intervention can be performed only at the next inspection date.

4.2.2.1 Maintenance actions

A maintenance action (decision) at timestep t_k for component i is denoted as a_k^i and might be imperfect as mentioned previously. Indeed, the quality of an imperfect maintenance (IM) action can be interpreted through the intervention gain which is defined as the difference between component's state before and after maintenance. The larger intervention gain is, the better maintenance quality can be obtained. From a practical point of view, maintenance quality may depend on several factors [109] such as spare part quality, repairman skills or constraints on limited repair duration, etc. Moreover, it is shown in literature that IM could have either random or deterministic effect on the condition state of a component after maintenance [28]:

- **IM with random quality**

  We consider the first case of IM in which the intervention gain is a random variable which also implies that state after maintenance of a component is also a random variable. For systems consisting of discrete-state components, state after maintenance of a component can be modeled by a probability law. Without loss of generality, it is assumed that state

of component i after maintenance at timestep t_k, denoted by $\bar{s}_k^i$, is distributed according to a probability mass function:

$$\sum_{j=0}^{u} Pr(\bar{s}_k^i = j | s_k^i = u) = 1 \text{ and } Pr(\bar{s}_k^i = j | s_k^i = u) \geq 0, \tag{4.3}$$

where u is the state before maintenance of component i. The intervention gain is then $d_k^i = (s_k^i - \bar{s}_k^i)$. In this work, in the case when IM has random quality, state after maintenance of a component is assumed to be obtained by uniformly discretely sampling from the interval from new state to its state before maintenance.

- **IM with deterministic quality**

  The second type of IM has deterministic effect on state after maintenance of a component. This means that an IM action can bring the maintained component to a specific state which is better than its state before maintenance. More precisely, $\bar{s}_k^i = s_k^i - d_k^i$. It is important to note that d_k^i must be less than or equal to s_k^i due to the fact that sate after maintenance of a component should not be negative.

4.2.2.2 Maintenance cost and economic dependence

Cost of separately maintaining component i is computed by using following equation:

$$c_k^i = c^{ins,i} + \mathbb{I}_k^{m,c,i} . \left(c^{s,sys} + c_k^{m,i} \right) \tag{4.4}$$

where:

- $c^{ins,i}$ is the cost induced by inspecting component i which must be paid even for survival components in order to reveal their current health conditions;
- $\mathbb{I}_k^{m,c,i}$ is the maintenance indicator of component i at timestep t_k whose value is equal to one if component i is maintained or equal to zero otherwise;
- $c^{s,sys}$ is the setup cost caused by, for instance, administrative handling or transportation of spare parts. It is assumed that the setup cost is the same for all components;
- $c_k^{m,i}$ is the component-specific maintenance cost which depends on the quality of maintenance action implemented on the component (imperfect maintenance). This cost is

calculated according to the following equation [28]:

$$c_k^{m,i} = c^{r,i}.\left(\frac{s_k^i - \bar{s}_k^i}{s_k^i}\right)^{\beta^i} \tag{4.5}$$

where:

- $c^{r,i}$ is a positive constant representing cost of replacing component i by a new one;
- $s_k^i > 0$ and $s_k^i \geq \bar{s}_k^i \geq 0$ are respectively the state before and after maintenance of component i at timestep t_k;
- β^i is a real positive number representing the imperfect maintenance characteristics of component i.

It should be noted that if component i is in new state ($s_k^i = 0$), it is unnecessary to maintain the component, i.e., the cost model in equation (4.5) is not needed and is got rid of by $\mathbb{I}_k^{m,c,i}$.

From a practical point of view, maintenance cost of a group of several components can be saved via setup cost sharing mechanisms thanks to the positive economic dependence between them [32]. For the system studied in this chapter, it is assumed that if several maintenance actions are grouped to carry out, the system setup cost has to paid only once. As a result, the maintenance cost at system level at timestep t_k denoted as c_k^{sys} is given by:

$$c_k^{sys} = \sum_{i=1}^{N} c_k^i - \mathbb{I}_k^{m,sys}.(N_k^{m,sys} - 1).c^{s,sys} \tag{4.6}$$

where:

- $N_k^{m,sys} = \sum_{i=1}^{N} \mathbb{I}_k^{m,c,i}$ is the number of maintained components at timestep t_k;
- $\mathbb{I}_k^{m,sys}$ is the system maintenance indicator at timestep t_k whose value is equal to one if there is at least one component being maintained or equal to zero otherwise;

4.3 AI-based maintenance optimization approach for multi-component systems

In this section, a maintenance planning approach is proposed for multi-component systems which integrates an ANN-based maintenance cost model and a system degradation model into the framework of MADRL to optimize maintenance decisions. An illustration of the proposed approach is depicted in Figure 4.1. Particularly, the first phase aims at learning a maintenance cost model at system level from collected condition monitoring data, and at modeling components' degradation process. The objective of the second phase is to construct an environment dedicated to reinforcement learning agents by employing the learned maintenance cost model and the degradation model from the first phase, and to train cooperative learning agents to optimize maintenance policies by letting them interact with the constructed environment.

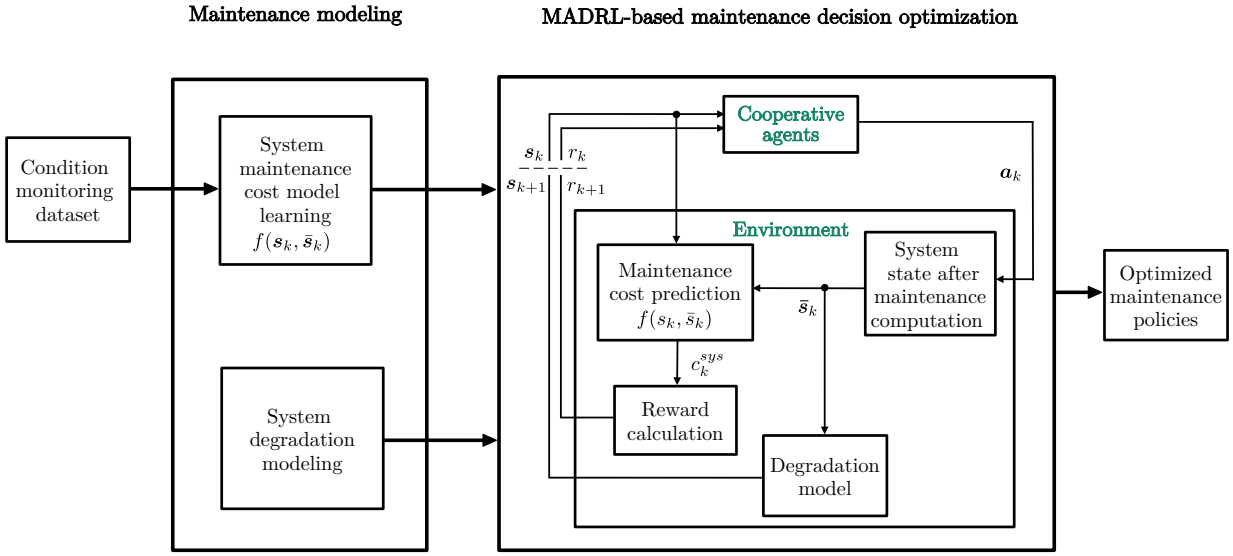


Figure 4.1: Illustration of the proposed AI-based maintenance optimization approach for multi-component systems.

In the following subsections, we present how to apply the proposed AI-based maintenance approach to optimize maintenance decisions of the system described in Section 4.2.

4.3.1 Maintenance modeling

4.3.1.1 Degradation model

The degradation model of the system presented in Section 4.2 are described through its components' own state transition matrix. Thanks to the assumption of independent degradation between components, the state transition matrix of component i ($i = 1, \ldots, N$) can be estimated by using the following equation:

$$\tilde{p}^i_{uv} = \frac{n^i_{uv}}{\sum_{v=0}^{m_i} n^i_{uv}} \tag{4.7}$$

where:

- $\tilde{p}^i_{uv}$ is the transition probability from state u to v of component i;
- n^i_{uv} is the total number of transitions from state u to v of component i.

However, this approach requires the availability of historical transition data which is very problematic in practice. To address this issue, several different approaches are proposed in the literature which are based on experts' judgment or analytical/numerical models [43]. In [110], judgment of experts combined with historical data is used to determine transition matrices which are then refined according to new coming data. Regression techniques are used in [111, 112] to compute transition matrices based on field data. An approach for discretizing stationary continuous-time continuous-state non-decreasing deterioration processes (e.g., gamma processes) into discrete-time discrete-state non-decreasing deterioration processes with stationary increments (e.g., discrete-time Markov chains) is presented in [113].

It should be noted that the components' degradation process of the system studied in this chapter are described by independent Markov processes. A novel model that can be used for modeling degradation interactions between components based on Markov processes is presented in Chapter 5.

4.3.1.2 System maintenance cost model learning

Traditionally, explicit cost structures at component level such as setup costs, spare part costs, maintenance labor costs, etc. are required to build a cost model at system level as illustrated in equation (4.6). In other words, the system-level maintenance cost is considered as a function of individual cost models at component level:

$$c^{sys} = f(\text{individual costs}) \tag{4.8}$$

From a practical point of view, maintenance actions are often grouped to implement in each maintenance intervention due to the component economic dependence leading to the fact that specific component-level cost data is not recorded separately, instead, only total cost is documented. Therefore, the availability requirement of separately collecting individual maintenance-related costs to construct the cost model at system level is not realistic. To address this issue, we propose an alternative view for computing the cost model of maintaining several components simultaneously representing the component economic dependence. In particular, the system-level maintenance cost will be considered as a function of components' degradation state:

$$c^{sys} = f(\text{components' health condition state}) \tag{4.9}$$

The system-level maintenance cost is then supposed to be learned by an ANN based on a dedicated CBM dataset. Indeed, this view is more in line with the data available in companies thanks to recent advances in sensing technologies allowing to collect rich degradation information. ANNs are chosen to learn maintenance cost models at system level due to their great approximating capacity (universal approximators). Moreover, recent advances dedicated for ANNs such as the invent of new activation functions, optimization algorithms to deal with vanishing gradient problem as well as fast development in parallel computational hardware (e.g., graphics processing unit, tensor processing unit) and software (e.g., TensorFlow, PyTorch framework) helps improve their scalability.

For the system studied in this chapter, its maintenance cost depends the quality of maintenance actions performed on its components. Therefore, we propose to use a multi-layer

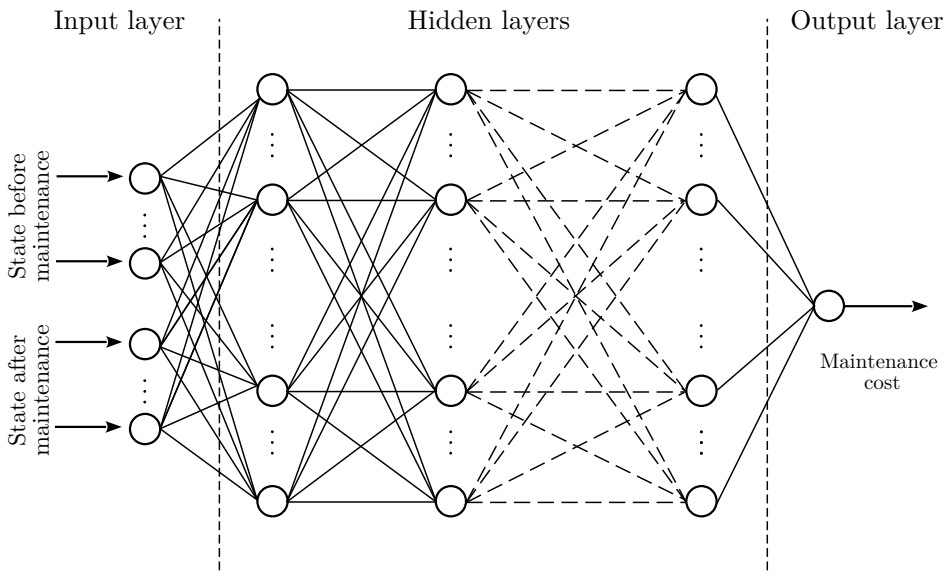


Figure 4.2: Illustration of the architecture of the ANN used to learn the maintenance cost model of the system studied in Chapter 4.

perceptron whose architecture is depicted in Figure 4.2, which takes as input an array consisting of the system state before and maintenance ($\boldsymbol{s}_k$ and $\bar{\boldsymbol{s}}_k$) and outputs the system maintenance cost (c_k^{sys}), to learn the maintenance cost model at system level of the studied system given that historical maintenance operations are collected in a dataset which has the form $\{\boldsymbol{s}_k, \bar{\boldsymbol{s}}_k, c_k^{sys}\}$ where:

- $\boldsymbol{s}_k = \begin{bmatrix} s_k^1 & s_k^2 & \dots & s_k^N \end{bmatrix}$ is the system state before maintenance at time t_k;
- $\bar{\boldsymbol{s}}_k = \begin{bmatrix} \bar{s}_k^1 & \bar{s}_k^2 & \dots & \bar{s}_k^N \end{bmatrix}$ is the system state after maintenance at time t_k. It should be noted that the maintenance duration is small in comparison with the inter-inspection time and then can be neglected;
- c_k^{sys} is the system maintenance cost at time t_k.

In addition, the maintenance dataset is assumed to be complete in the sense that it consists of all state transitions of the maintained system, balanced and is large enough to be applicable for the learning process of ANNs with high approximation accuracy. It should be noted that

the hyperparameters used for training such as the number of epochs, the number of hidden layers and the number of neurons per each hidden layer need to be carefully chosen to avoid the case of underfitting and overfitting. Furthermore, ADAM algorithm [114] which is an extension of stochastic gradient descent is highly recommended to be used as the optimizer for training due to its accuracy and time-efficiency.

4.3.2 MADRL-based maintenance decision optimization

Maintenance decision-making problem of the multi-component system described in Section 4.2 is modeled as a cooperative multi-agent task with a group of N agents, in which one agent controls maintenance decisions of one component (one might use one agent as a controller of a sub-system, however, the number of components in a sub-system should not be too large), and can observe degradation states of other components in the system (fully observable setting).

4.3.2.1 Agent-environment interface

The interaction between multiple maintenance decision-makers and the maintained system is modeled by a MMDP whose main elements' formal definition are presented in the following:

State space A component i has its own health condition state space $\mathcal{S}^i$ that helps to form the joint state space $\mathcal{S} = \mathcal{S}^1 \times \mathcal{S}^2 \times ... \times \mathcal{S}^N$. As a result, the joint (system) state at timestep t_k is a vector consisting all component states defined as $\boldsymbol{s}_k = [s_k^1, s_k^2, \ldots, s_k^N]$ where s_k^i is the degradation level of component i at that inspection time.

Action space Similarly, each agent $AG^i \in \mathcal{AG}$ has its own action space $\mathcal{A}^i$ which is a set of all maintenance actions available for agent i that helps to form the joint action space $\mathcal{A} = \mathcal{A}^1 \times \mathcal{A}^2 \times ... \times \mathcal{A}^N$. Therefore, the joint maintenance action at timestep t_k is a vector composed of all maintenance actions chosen individually by all agents denoted as $\boldsymbol{a}_k = [a_k^1, a_k^2, \ldots, a_k^N]$.

It should be noted that there are 3 actions available to be chosen for an agent $AG^i \in \mathcal{AG}$ at

any timestep in the case of random IM quality, which are:

$$a_k^i = \begin{cases} 0 & \text{leave component } i \text{ as it is} \\ 1 & \text{implement IM on component } i \\ 2 & \text{replace component } i \end{cases} \tag{4.10}$$

In the case where IM has deterministic quality, there are (m^i+1) possible maintenance levels that can be performed on component i. The encoded number of a maintenance action is indeed the value of the corresponding intervention gain:

$$a_k^i = \begin{cases} 0 & \text{leave component } i \text{ as it is} \\ j & \text{implement IM level } j \text{ on component } i \ (1 \leq j \leq m^i - 1) \\ m^i & \text{replace component } i \end{cases} \tag{4.11}$$

Transition function The probability that a system state after maintenance ($\bar{\boldsymbol{s}}_k$) at timestep t_k degrades to a system state before maintenance ($\boldsymbol{s}_{k+1}$) at next timestep t_{k+1} is characterized by the transition matrices $\tilde{\boldsymbol{P}}^i$ $(i = 1, \ldots, N)$.

Reward function Maintenance decision optimization is usually associated with the problem of balancing the trade-off between maintenance frequency and unplanned downtime cost in the sense that if maintenance actions are carried out more often, maintenance cost might be high, or if maintenance operations are conducted less frequently, system breakdowns occur more recurrently leading to negative consequences. To take this issue into consideration, the reward generator is defined as a function of the maintenance cost at system level and the downtime cost which is given by:

$$\begin{aligned} r_k &= -c_k^{sys} - \mathbb{I}_k^{dt}.c^{dt} \\ &= -f(\boldsymbol{s}_k, \bar{\boldsymbol{s}}_k) - \mathbb{I}_k^{dt}.c^{dt} \end{aligned} \tag{4.12}$$

where:

- $c_k^{sys} = f(\boldsymbol{s}_k, \bar{\boldsymbol{s}}_k)$ is the system maintenance cost predicted by the trained ANN;

- $\mathbb{I}_k^{dt}$ is the system status indicator at timestep t_k whose value is equal to one if the system is in failed state or is equal to zero otherwise;
- c^{dt} is a constant representing the downtime cost.

4.3.2.2 Agent training process

Next step is to train cooperative agents to optimize maintenance policies by letting them interact with the environment constructed in previous section. The interaction between learning agents and the system being maintained is depicted in Figure 4.3.

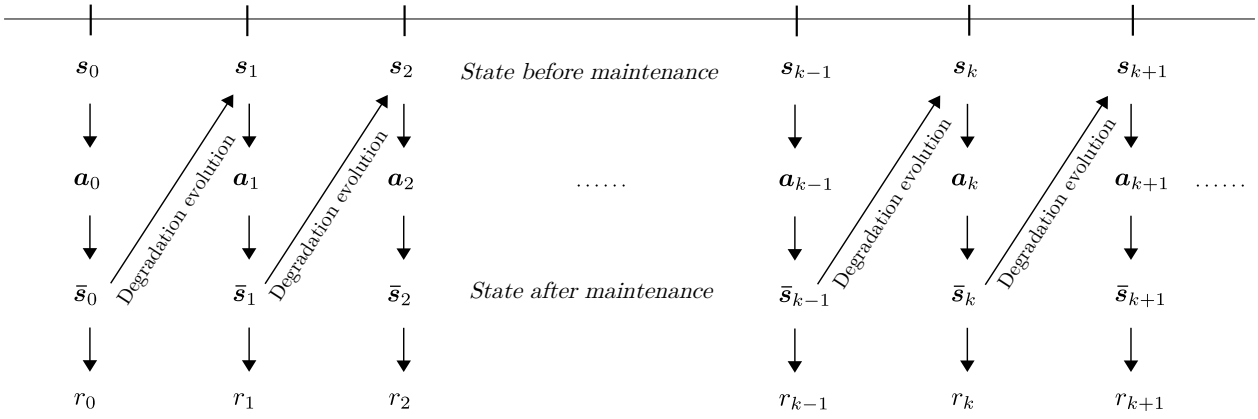


Figure 4.3: Illustration of a sequential maintenance decision-making process.

Particularly, at each decision time t_k, each agent $AG^i \in \mathcal{AG}$ observes a system state $\boldsymbol{s}_k \in \mathcal{S}$ and then chooses an action a_k^i from its own action space $\mathcal{A}^i$ based on its own policy $\pi^i(a_k^i|\boldsymbol{s}_k)$. The individual actions form a joint action $\boldsymbol{a}_k \in \mathcal{A}$. Once the chosen joint maintenance action is carried out, the system transitions from its state before maintenance $\boldsymbol{s}_k \in \mathcal{S}$ to a state after maintenance $\bar{\boldsymbol{s}}_k \in \mathcal{S}$ and releases a scalar reward $r_k \in \mathbb{R}$ shared by all agents. After that, the maintained system naturally degrades to a next state before maintenance $\boldsymbol{s}_{k+1} \in \mathcal{S}$ at next inspection time t_{k+1}.

The objective of cooperative agents is to maximize the expectation of discounted return defined in Chapter 3 which can be realized by the customized WQMIX algorithm presented in Section 3.3.3.2.

4.4 Numerical experiments

To investigate the performance of the proposed AI-based maintenance approach, two multi-component systems are studied in this section. Firstly, an experiment is conducted on a 4-component system in Subsection 4.4.1 to illustrate how to apply the proposed approach for maintenance optimization considering imperfect maintenance actions with both deterministic and random quality. Different configurations of this small system are then investigated to examine the impact of system structure on maintenance polices optimized by the customized WQMIX. Finally, a study conducted on a 15-component system is presented in Subsection 4.4.2 with the objective to examine the scalability of the proposed maintenance approach.

4.4.1 Four-component system

The studied 4-component system has parallel structure as illustrated in Figure 4.4. Each component is assumed to have 5 discrete health condition states ranging from new to complete failure. The degradation of the components are characterized by the following transition matrices:

$$\tilde{\boldsymbol{P}}^1 = \begin{bmatrix} 0.3 & 0.3 & 0.2 & 0.15 & 0.05 \\ 0 & 0.2 & 0.3 & 0.3 & 0.2 \\ 0 & 0 & 0.3 & 0.4 & 0.3 \\ 0 & 0 & 0 & 0.4 & 0.6 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tilde{\boldsymbol{P}}^2 = \begin{bmatrix} 0.1 & 0.3 & 0.3 & 0.2 & 0.1 \\ 0 & 0.1 & 0.3 & 0.3 & 0.3 \\ 0 & 0 & 0.3 & 0.3 & 0.4 \\ 0 & 0 & 0 & 0.2 & 0.8 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.13}$$

$$\tilde{\boldsymbol{P}}^3 = \begin{bmatrix} 0.25 & 0.3 & 0.2 & 0.2 & 0.05 \\ 0 & 0.1 & 0.2 & 0.3 & 0.4 \\ 0 & 0 & 0.2 & 0.3 & 0.5 \\ 0 & 0 & 0 & 0.2 & 0.8 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tilde{\boldsymbol{P}}^4 = \begin{bmatrix} 0.2 & 0.2 & 0.3 & 0.3 & 0 \\ 0 & 0.1 & 0.3 & 0.3 & 0.3 \\ 0 & 0 & 0.2 & 0.3 & 0.5 \\ 0 & 0 & 0 & 0.3 & 0.7 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.14}$$

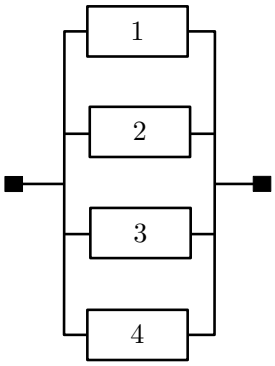


Figure 4.4: Illustration of the 4-component parallel system studied in Chapter 4.

4.4.1.1 Maintenance cost model learning

It should be noted that the maintenance cost model at system level of the studied system, which is traditionally constructed based on the individual cost models at component level, is not given at hand. Instead, we must learn such cost model from a historical dataset of system states before and after maintenance with corresponding system maintenance costs.

Table 4.1: Illustration of the simulated dataset used for the learning of the system-level maintenance cost model of the 4-component system studied in Chapter 4.

k	$\boldsymbol{s}_k$	$\bar{\boldsymbol{s}}_k$	c_k^{sys}
0	[0 0 0 0]	[0 0 0 0]	0.00
1	[3 1 2 0]	[2 1 1 0]	141.59
2	[2 2 4 2]	[2 1 4 0]	185.25
3	[3 4 4 0]	[2 0 0 0]	231.59
4	[3 3 3 2]	[2 0 3 0]	236.59
5	[4 3 4 3]	[0 1 4 3]	212.88
6	[4 1 4 4]	[0 1 0 0]	289.00
7	[1 2 3 0]	[0 1 3 0]	200.25
8	[0 4 4 2]	[0 4 0 1]	167.43
.....			

Table 4.1: Illustration of the simulated dataset used for the learning of the system-level maintenance cost model of the 4-component system studied in Chapter 4.

k	$\boldsymbol{s}_k$	$\bar{\boldsymbol{s}}_k$	c_k^{sys}
2993	[4 3 4 3]	[4 1 0 1]	203.75
2994	[4 4 4 2]	[4 0 4 1]	182.43
2995	[4 1 4 3]	[4 0 0 1]	239.86
2996	[4 2 1 4]	[0 0 0 4]	299.00
2997	[0 3 4 4]	[0 1 4 0]	197.89
2998	[0 3 4 3]	[0 0 0 1]	239.86
2999	[2 1 0 3]	[1 0 0 1]	198.61

In this study, a dataset consisting of 3000 sample points depicted in Table 4.1 is used to learn the maintenance cost model of the studied 4-component system by using an feed forward neural network whose architecture is depicted in Figure 4.2 which contains two hidden layers of 100 neurons. The parameters used for the data generation process are given in Appendix A.1.

Each run of the training contains 500 epochs and the used batch size is 32. The learning rate is set to 0.0025. Mean squared loss is used to assess the model's performance. The training process is conducted 30 times to examine its robustness as well as to access its variability. The average of the obtained mean squared losses on training and validation set are respectively 0.22 and 0.62 with the corresponding standard of deviations are 0.11 and 0.38. Figure 4.5 shows the convergence of mean squared losses during one run. It can be noticed that the mean squared losses and their standard of deviation are quite small in comparison with the range of maintenance costs observed from the dataset. Therefore, we can conclude that the maintenance cost model at system level of the studied 4-component system is learned by the ANN.

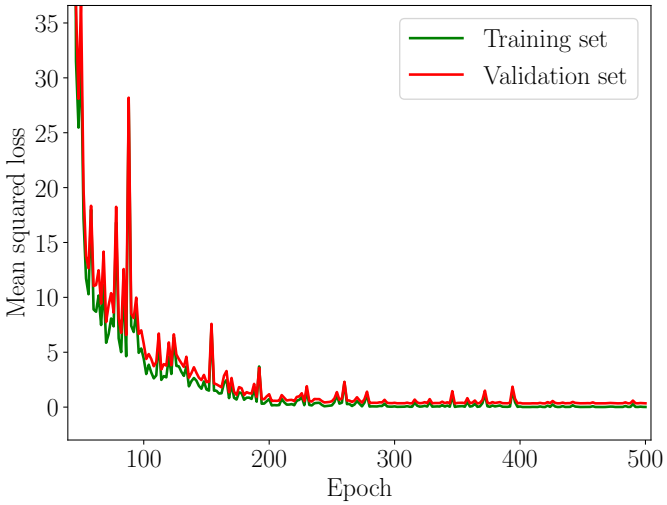


Figure 4.5: The convergence of mean squared loss on training and validation set.

4.4.1.2 Maintenance optimization

In this section, the customized WQMIX algorithm for fully observable setting presented in Section 3.3.3.2 is implemented to optimize maintenance actions of the 4-component parallel system.

Table 4.2: Hyperparameters of the branching networks used for the simulation experiments conducted in Chapter 4.

	4-component system	15-component system
Number of inputs	4	15
Number of outputs in each branch	3 or 5	3 or 5
Number of neuron in the shared MLP	[128, 128]	[256, 256]
Number of neuron the value MLP	[128]	[128]
Number of neuron the advantage MLP	[64]	[128]

The hyperparameters of the branching and mixing network used for representing the agents' action-value function are presented in Table 4.2 and 4.3. It should be noted that the number

of outputs in each branch of the branching network is 5 and 3 in the case of considering deterministic and random imperfect maintenance quality respectively.

Table 4.3: Hyperparameters of the mixing networks used for the simulation experiments conducted in chapter 4.

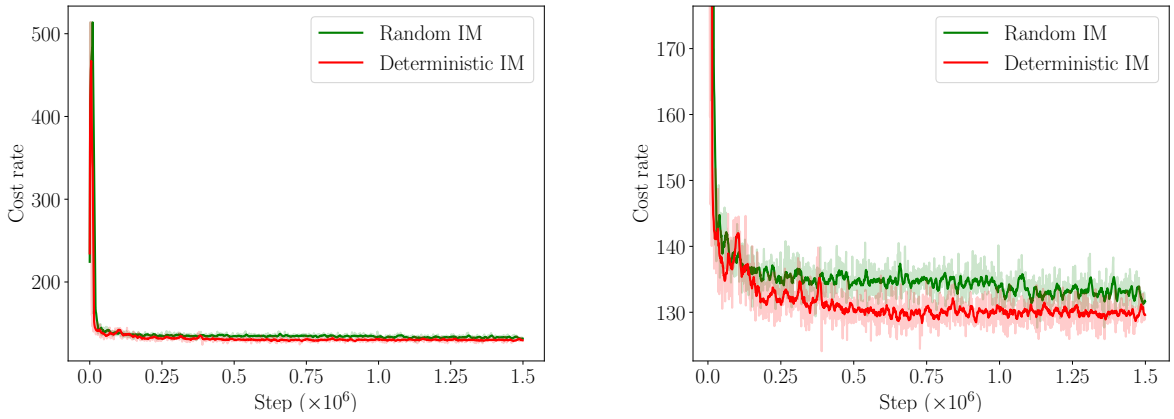
	4-component system	15-component system
Number of inputs	4	15
Number of outputs	1	1
Number of neurons in hidden layers of $\boldsymbol{Q}$	[64, 64]	[128, 128]
Number of neurons in hidden layers of $\boldsymbol{Q}^{tot}$	[64, 64]	[128, 128]

Cooperative learning agents are trained through 1.5×10^6 steps to optimize maintenance decisions of the studied system. Learning rates are scheduled to linearly declined from 0.008 to 0.00025 during the first 500×10^3 training steps. The discount factor, the mini-batch size and the size of replay buffer are 0.99, 128, and 300×10^3 respectively. The target update frequency is 5×10^3 steps. In order to supervise the training, latest policy networks representing $\boldsymbol{Q}^{tot}$ at every 2×10^3 steps are employed to interact with validation environments 5×10^3 times and we then observe corresponding cost rates.

The results of maintenance optimization processes are depicted in Figure 4.6 with the best found cost rates are 132.48 and 129.74 in the case where imperfect maintenance has random and deterministic quality respectively.

4.4.1.3 Sensitivity analysis to system structure

The assumption that the joint action-value function can be monotonically decomposed into the local ones across agents in theory affects the performance of the customized WQMIX algorithm when it is applied to optimize maintenance decisions for multi-component systems. Particularly, the nonlinearity of the reward function used to define the maintenance planning objective often lies on the economic dependence and downtime cost model as illustrated in equation (4.12). Therefore, if the economic dependence model between maintained compo-



(a) The evolution of cost rates during training. (b) A closer look at the evolution of cost rates.

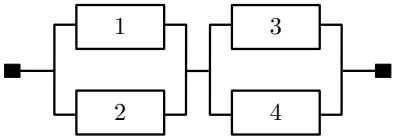
Figure 4.6: Agent training monitoring for maintenance optimization of the 4-component parallel system studied in Chapter 4.

nents in a system is complicated or the downtime cost model is complex, the monotonic factorization scheme might not be able to fully represent the function class to which the true joint action-value function belongs. As a result, the performance of the customized WQMIX might be declined for such a system.

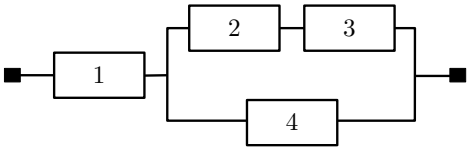
For the systems studied in this chapter, the economic dependence between components is represented by a setup cost saving mechanism which is fixed as presented in Section 4.2.2.2. However, the downtime cost model is indeed a function of the system structure from a reliability block diagram point of view, which can be changed. Therefore, in order to investigate this insight, numerical experiments are conducted on a 4-component system with different structure configurations (e.g., series, parallel or mixed). We refer the parallel system depicted in Figure 4.4 as the "configuration 1" and the reliability block diagram of three other configurations is given in Figure 4.7.

Thanks to the small number of components of the studied system, VI algorithm can be implemented to find exact solutions which are used as baselines for evaluating the customized WQMIX's performance. The optimization processes terminate if the estimation accuracy of

(a) Configuration 2



(b) Configuration 3



(c) Configuration 4

Figure 4.7: Three different structure configurations of the 4-component system studied in Chapter 4.

all system state values is less than or equal to 0.001. The obtained policies are evaluated via simulation results through 5 runs of 10^4 maintenance interventions.

Table 4.4: Comparison of optimized cost rates in the case where imperfect maintenance has random quality.

	Configuration 1	Configuration 2	Configuration 3	Configuration 4
WQMIX	132.48	398.67	214.66	222.29
VI	132.48	398.67	214.66	222.29

The results of this case study are provided in Table 4.4 and 4.5 showing that the customized WQMIX is capable of approaching exact solutions provided by VI for all configurations in the case of random imperfect maintenance quality. However, sub-optimal policies are obtained when imperfect maintenance quality is deterministic. This can be explained by the fact that there are too many deterministic maintenance actions that have similar effect at a given

Table 4.5: Comparison of optimized cost rates in the case where imperfect maintenance has deterministic quality.

	Configuration 1	Configuration 2	Configuration 3	Configuration 4
WQMIX	129.74	398.67	222.34	222.29
VI	128.86	398.67	211.52	222.29

system state causing difficulties for decision-making agents to distinguish between them.

Based on the obtained results, it can be noticed that, though the monotonic decomposition scheme used for the joint action-value function in theory might not fully represent the ground-truth function class to which it belongs, the ability of fully observing system states when making decisions and the weight sharing mechanism in the branching network are believed to help the customized WQMIX algorithm to find near-optimal/optimal maintenance polices.

4.4.2 Fifteen-component system

In this section, we examine the performance of the customized WQMIX when it is applied to optimize maintenance of a large-scale series-parallel system composed of 15-components. The structure of the studied system is depicted in Figure 4.8, which is divided into five subsystems of identical components. Particularly, component 1 is considered as the first subsystem. Component 2 and 3 together form the next subsystem. The third one is composed of component 4, 5 and 6. The fourth subsystem contains component 7, 8, 9 and 10. Finally, the last subsystem is comprised of component 11 upto 15.

The components of subsystem 1, 2, 3 and 4 respectively degrade according to the transition matrices $\tilde{\boldsymbol{P}}^1$, $\tilde{\boldsymbol{P}}^2$, $\tilde{\boldsymbol{P}}^3$ and $\tilde{\boldsymbol{P}}^4$ as expressed in equation (4.13) and (4.14). The degradation of the components of the last subsystem is characterized by the transition matrix expressed in equation (4.15). The maintenance-related cost parameters of the system is presented in

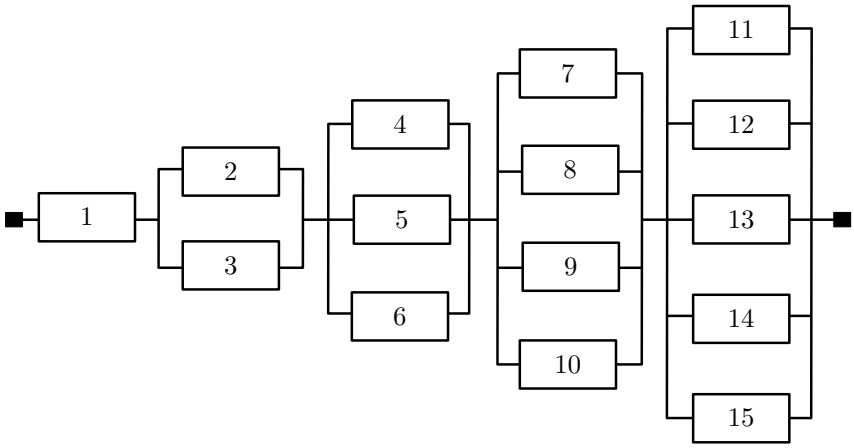


Figure 4.8: Reliability block diagram of the 15-component system studied in Chapter 4.

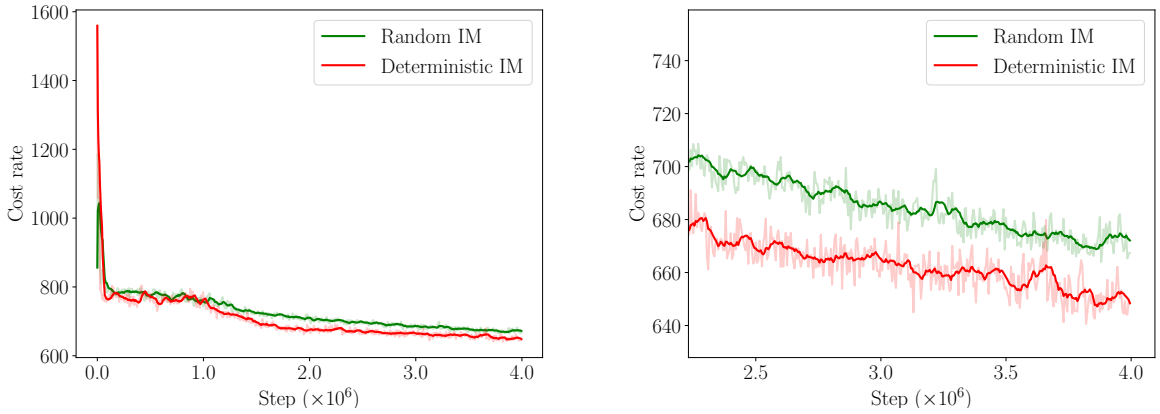
appendix A.2.

$$\begin{bmatrix} 0.25 & 0.2 & 0.3 & 0.2 & 0.05 \\ 0 & 0.15 & 0.2 & 0.4 & 0.25 \\ 0 & 0 & 0.2 & 0.4 & 0.4 \\ 0 & 0 & 0 & 0.2 & 0.8 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.15}$$

VI algorithm is not applicable here due to the high dimensionality of state and action space of the studied system. Particularly, the total number of system states is greater than 30×10^9, and the number of all possible system actions is more than 30×10^9 and 14×10^6 in the situation where IM has deterministic and random quality respectively. These numbers indicate the cure of dimensionality that optimization algorithms must face when they are applied to optimize maintenance decisions of large-scale multi-component systems.

In this study, RL agents are trained through 4×10^6 steps with a larger branching and mixing network in comparison to the ones used for the 4-component system as presented in Table 4.2 and 4.3. Learning rates are scheduled to decline linearly from 0.008 to 0.00025 after 500×10^3 training steps. The discount factor, the mini-batch size and the size of replay buffer are respectively 0.99, 128, and 10^6. The target update frequency is 20×10^3 steps. The latest policy networks after every 5×10^3 steps are employed to interact with validation

environments 10^4 times to compute corresponding cost rates.



(a) The evolution of cost rates during training (b) A closer look at the evolution of cost rates

Figure 4.9: Agent training monitoring for maintenance optimization of the 15-component system studied in Chapter 4.

The evolution of cost rates during training is illustrated in Figure 4.9 with the best found cost rates are 673.14 and 651.23 in the case where IM has random and deterministic quality respectively. An illustration of the optimized direct-mapping maintenance polices is depicted in Table 4.6 showing that the components of the studied system are often grouped to maintain due to the economic dependence between them. One can notice that component 1, that is the most important component from a reliability block diagram point of view, is replaced very frequently in order to avoid the system downtime. Particularly, it is replaced starting from degradation level 1. As a reminder, the numbers encoded maintenance actions in the second column (IM with random quality) are: 0 ("do nothing"), 1 ("imperfect maintenance") and 2 ("replacement"), while a number j ($j = 0, .., 4$) in the third column (IM with deterministic quality) corresponds to maintenance intervention gain level j. Moreover, it can also be seen that IM actions are carried out more regularly in the case where they have deterministic quality especially for the last subsystem whose IM constant is high implying that the cost of implementing an IM action is much less expensive than the one of carrying out a replacement action.

Table 4.6: Illustration of maintenance actions optimized for the 15-component system studied in Chapter 4.

	Optimized maintenance actions	
Components' state	IM with random quality	IM with deterministic quality
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]	[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]	[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 1 4 0 0 0 3 2 2 3 1 1 3 3 2]	[0 0 2 0 0 0 2 0 0 0 0 0 2 0 0]	[0 0 4 0 0 0 1 1 2 0 1 1 1 1 1]
[0 3 3 1 1 2 3 4 4 3 1 2 2 4 3]	[0 2 2 2 0 0 2 0 0 0 0 0 0 0 0]	[0 3 3 0 0 2 1 0 4 0 0 0 0 0 0]
[3 0 1 1 2 4 3 4 4 4 3 2 2 4 3]	[2 0 0 2 0 0 1 0 2 2 0 0 2 0 0]	[3 0 0 0 0 4 2 0 4 0 1 1 0 0 1]
[1 1 2 3 4 4 2 4 0 3 4 4 0 4 4]	[2 0 0 2 0 0 2 0 0 0 0 0 0 2 0]	[1 0 0 0 4 4 0 0 0 0 4 4 0 0 4]
[2 3 3 2 4 4 3 4 0 4 4 4 3 2 4]	[2 2 2 2 0 0 1 0 0 0 0 0 2 0 0]	[2 3 3 2 4 4 1 0 0 0 0 4 1 1 0]
[3 0 2 2 4 4 1 4 2 4 4 4 0 2 4]	[2 0 0 1 2 2 2 0 0 0 0 0 0 0 0]	[3 0 2 0 4 4 0 0 2 0 4 4 0 1 0]
[0 4 2 2 1 2 3 4 4 4 4 4 3 3 4]	[0 2 0 0 0 0 2 0 0 2 0 0 2 0 0]	[0 4 1 0 0 2 2 0 4 0 0 0 1 1 4]
[0 3 3 2 2 4 3 4 4 0 4 4 2 3 4]	[0 2 1 2 0 0 1 0 0 0 0 0 2 0 0]	[0 3 3 0 0 4 2 0 4 0 4 4 2 0 4]
[2 1 4 3 3 4 0 4 4 0 4 4 2 4 4]	[2 0 2 2 0 2 0 0 0 0 0 0 1 0 2]	[2 0 4 3 0 4 0 0 4 0 4 4 0 4 0]
[0 2 2 3 3 0 2 4 4 2 4 4 3 4 2]	[0 2 0 2 0 0 2 0 0 1 0 0 2 0 0]	[0 0 2 3 0 0 1 0 4 0 4 4 1 0 1]
[0 1 2 1 4 1 0 4 4 3 4 4 3 4 3]	[0 0 2 2 0 0 0 0 0 0 0 0 2 0 1]	[0 0 0 0 4 0 0 0 0 0 4 4 1 0 1]
[1 1 3 1 4 4 0 4 4 4 4 4 1 4 4]	[2 0 2 2 0 0 0 0 0 0 0 0 0 0 2]	[1 0 3 0 4 4 0 0 0 0 4 4 1 0 0]
[2 4 2 2 4 4 3 4 4 4 4 4 3 4 2]	[2 2 1 2 0 2 2 0 0 0 0 0 2 0 0]	[2 4 1 0 4 4 1 0 4 0 4 4 0 0 1]
[3 3 4 3 4 1 1 4 4 4 4 4 2 4 4]	[2 2 2 2 0 0 2 0 0 0 0 0 2 2 0]	[3 3 4 0 4 0 1 0 4 0 4 0 0 0 4]
[2 2 3 0 4 1 3 4 4 4 4 4 4 3 4]	[2 0 2 0 0 0 2 0 0 0 0 0 0 2 2]	[2 0 3 0 4 0 2 0 4 0 4 0 0 1 4]
[1 4 1 3 4 4 1 4 4 4 4 4 4 2 2]	[2 2 0 2 0 2 2 0 0 0 0 0 2 0 0]	[1 4 1 0 4 4 0 0 4 0 4 4 0 1 1]
[2 1 4 2 4 3 2 4 4 4 4 4 2 2 2]	[2 0 2 2 2 2 2 0 0 0 0 0 2 0 0]	[2 0 4 0 4 3 1 0 4 0 4 4 1 1 0]
[0 4 2 2 0 3 3 4 4 4 4 4 3 4 3]	[0 2 2 0 0 0 1 0 0 2 2 0 2 0 0]	[0 4 0 0 0 3 2 0 4 0 0 4 1 0 3]

4.5 Conclusions

In this chapter, a maintenance optimization approach based on the framework of MADRL is proposed for systems consisting of multiple components taking into consideration the scientific issues related to the requirement of individual cost models at component level to construct maintenance cost models at system level.

Numerical studies are conducted on a small 4-component system with different configurations and on a 15-component series-parallel system considering both random and deterministic imperfect maintenance actions to examine the performance as well as the scalability of the proposed maintenance approach. The obtained simulation results show that ANNs are suitable to system maintenance cost forecasting and MADRL algorithms effectively optimize maintenance decisions.

Chapter 5

Integrating a novel degradation interaction model and partial observability into the AI-based maintenance approach

5.1 Introduction

This chapter aims to present the second contribution of this PhD which allows to model the stochastic dependence in systems consisting of discrete-state components. Particularly, a novel state-rate interaction model, which implies that the degradation state of a component might affect the reliability of other components, is proposed to describe the component stochastic dependence.

The maintenance optimization approach presented in previous chapter is applied to plan maintenance operations of the systems studied in this chapter in both fully and partially observable setting. Specifically, the maintenance approach integrates a the proposed degradation interaction model and a maintenance cost model of the system under consideration into the framework of MADRL to optimize maintenance decisions. It should be noted that the system cost model can be learned by an ANN if dedicated condition monitoring data is available as presented in Section 4.3.1.2.

The chapter is structured as follows. Section 5.2 presents the proposed component stochastic dependence model. A presentation of maintenance decision optimization processes based on MADRL in partially and fully observable setting is given in Section 5.3. Numerical experiments are depicted and analyzed in Section 5.4 to investigate impacts of component dependencies and agent's observability on optimized maintenance policies. Finally, the last section concludes the chapter and gives some perspectives.

5.2 Component degradation interactions

We consider a series-parallel system consisting of N components which are grouped into N^{sub} subsystems as illustrated in Figure 5.1. Subsystem j has $N^{sub,j}$ components of the same type j. As a result, $N = \sum_{j=1}^{N^{sub}} N^{sub,j}$. A component i at a periodic inspection time t_k can be observed in any discrete health condition state, $s_k^i \in \{0, ..., m^i\}$, ranging from new state to complete failure. It is assumed that in the absence of maintenance interventions and degradation interactions between components, the state transition of a component i between two consecutive inspections obeys its inherent Markov transition matrix:

$$\tilde{\boldsymbol{P}}^i = \begin{bmatrix} \tilde{p}_{00}^i & \tilde{p}_{01}^i & \tilde{p}_{02}^i & \cdots & \tilde{p}_{0m^i}^i \\ 0 & \tilde{p}_{11}^i & \tilde{p}_{12}^i & \cdots & \tilde{p}_{1m^i}^i \\ 0 & 0 & \tilde{p}_{22}^i & \cdots & \tilde{p}_{2m^i}^i \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix} \tag{5.1}$$

in which $\tilde{p}_{uv}^i$ is a non-negative real number representing the intrinsic degradation transition probability from state u to v of component i that satisfies: $\sum_{v=u}^{m^i} \tilde{p}_{uv}^i = 1, \forall u \in \{0, 1, \ldots, m^i\}$. It should be noted that if component i and i' belong to the same subsystem (are of the same type), then $\tilde{\boldsymbol{P}}^i = \tilde{\boldsymbol{P}}^{i'}$.

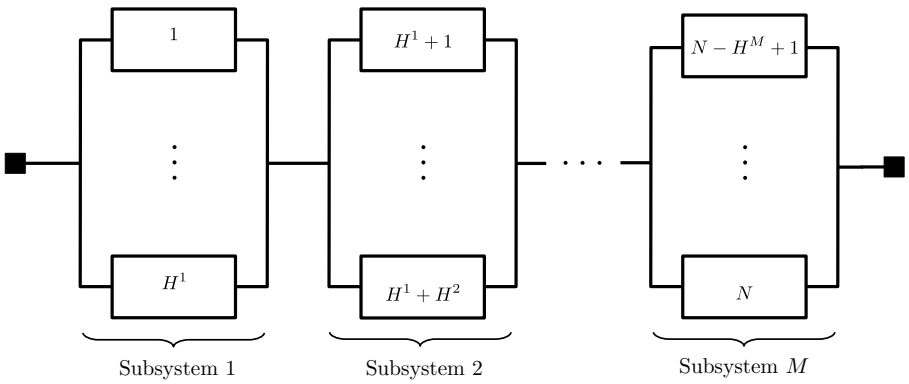


Figure 5.1: Illustration of a series-parallel system consisting of M subsystems in which subsystem j is composed of H^j components of the same type ($j = 1, 2, ..., M$).

Once the stochastic dependence is taken into consideration, component i degrades according to a new transition probability matrix denoted as $\boldsymbol{P}^i = \{p^i_{uv}\}_{(m^i+1)\times(m^i+1)}$ that is different from $\tilde{\boldsymbol{P}}^i$. In particular, we consider in this chapter the case where condition states of a component can accelerate the degradation speed of other components. Within the framework of Markov processes, this can be modeled by decreasing probabilities of staying in current states of the component, while increasing its probabilities of transitioning to worse conditions. In that way, we propose the following model:

$$p^i_{uv} = \tilde{p}^i_{uv} + \Delta^i_{uv} \tag{5.2}$$

where:

- $\tilde{p}^i_{uv}$ is the intrinsic degradation transition probability from state u to v of component i;
- p^i_{uv} is the degradation transition probability from state u to v of component i considering the stochastic dependence impacts from other components;
- Δ^i_{uv} indicates the change in the transition probability of component i due to the component stochastic dependence and verifies the following proprieties:

$$\begin{cases} -\tilde{p}^i_{uu} \leq \Delta^i_{uu} < 0 & \text{if } u = v \text{ and } u \neq m^i \\ 0 < \Delta^i_{uv} \leq 1 - \tilde{p}^i_{uv} & \text{if } v > u \text{ and } u \neq m^i \\ \Delta^i_{uv} = 0 & \text{otherwise} \end{cases} \tag{5.3}$$

  Moreover, as the sum of all transition probabilities from a given state should be equal to 1. From equation (5.2) we get:

$$\sum_{v=u}^{m^i} \Delta^i_{uv} = 0 \text{ or } \Delta^i_{uu} = -\sum_{v=u+1}^{m^i} \Delta^i_{uv} \tag{5.4}$$

The transition probability variation Δ^i_{uv} may depend on the current degradation speed of component i which is determined by the index u and v in the transition matrix as well as on other components' current state. In that way, it can be expressed as belows:

$$\Delta^i_{uv} = \xi^i_{uv} . \sum_{j=1}^{N} \tau^{ij} \tag{5.5}$$

where:

- ξ_{uv}^i is a constant term representing the maximal marginal change in the original transition probability from state u to v of component i. From a practical point of view, the more the state degrades, the higher the change on the transition probability. Hence, it is reasonable to suggest the following model for ξ_{uv}^i based on equation (5.3) and (5.4):

$$\xi_{uv}^i = \begin{cases} -\tilde{p}_{uu}^i & \text{if } v = u \text{ and } u \neq m^i \\ \frac{\tilde{p}_{uu}^i}{1-\tilde{p}_{uu}^i}\tilde{p}_{uv}^i & \text{if } v > u \text{ and } u \neq m^i \\ 0 & \text{otherwise} \end{cases} \tag{5.6}$$

- τ^{ij} represents the degradation interaction impact of component j on component i. It may depends on both the current state s^j of component j and the dependence level between component i and j. In that way, we suggest a general model as follows:

$$\tau^{ij} = \zeta^{ij}\left(\frac{s^j}{m^j}\right)^{\alpha^j} \tag{5.7}$$

  in which:

  - $0 \leq \zeta^{ij} \leq 1$ is a non-negative real number quantifying the stochastic dependence level of component i on component j. The higher ζ^{ij}, the more component i is dependent on component i stochastically. $\zeta^{ij} = 0$ when $i = j$ or components i and j degrade independently. For example, the component stochastic dependence may only exits within a subsystem of series-parallel systems due to the sharing of a common load [115], i.e., $\zeta^{ij} > 0$ if components i and j belong to the same subsystem and $\zeta^{ij} = 0$ if components i and j are of two different subsystems. To characterize the stochastic dependence between components, we denote $\boldsymbol{Z} = \{\zeta^{ij}\}_{N\times N}$ the degradation interaction matrix.

  - α^j is a non-negative constant describing the inherent state-rate interaction of component j. When components i and j are stochastically dependent ($\zeta^{ij} > 0$), $\alpha^j = 0$ means that the stochastic dependence between components i and j does not depend on the current state of component j. When $\zeta^{ij} > 0$ and $\alpha^j > 0$, the inherent state-rate interaction impact is proportional with the degradation level (state) of

component j. As an example, it is shown that the degradation interactions between two components of an industrial cool box system only occurs if one gas tube is already subject to high fouling state (high degradation state) leading to the excess-heated gas be forced to go through the other one which then leads to accelerated fouling as a result [69]. Furthermore, if component j is in a given surviving state, i.e., $0 < s^j/m^j < 1$, then, the higher α^j is, the lower the inherent state-rate interaction impact. The later also implies that when α^j is very high, the degradation interactions between component i and j is only triggered by the failure of component j. It should be noticed that the proposed stochastic interaction model in this case becomes an existing stochastic dependence model: the failure of a component affects the degradation process of other components [23, 24].

The components' inherent transition probability matrix can be estimated from data collected when they operate individually or might be provided by their ordinary equipment manufacturer. The parameters representing degradation interactions between components, which depend on the system's characteristics such as design structure, operating conditions, component types should be estimated from testing or historical data. If historical degradation data is available, one can use particle filters to estimate those parameters which allow for an online numerical estimation of the parameter values by means of a recursive Bayesian inference approach [20]. Due to the lack of real data, we assume in this study that the degradation-interaction-related parameters and the inherent state transition matrices are known.

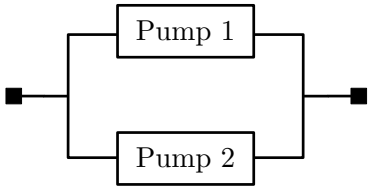


Figure 5.2: Reliability block diagram of a parallel system consisting of two identical pumps.

To illustrate the proposed degradation interaction model, we consider a parallel system con-

sisting of two identical pumps as shown in Figure 5.2 in which the health condition states of each pump are encoded as **0**: "as good as new state", **1**: "minor degradation", **2**: "severe degradation" and **3**: "failed state". It is assumed that the inherent transition matrix of each pump is given as follows:

$$\tilde{\boldsymbol{P}}^1 = \tilde{\boldsymbol{P}}^2 = \begin{bmatrix} 0.80 & 0.10 & 0.05 & 0.05 \\ 0.00 & 0.80 & 0.10 & 0.10 \\ 0.00 & 0.00 & 0.80 & 0.20 \\ 0.00 & 0.00 & 0.00 & 1.00 \end{bmatrix} \tag{5.8}$$

It is also assumed that the pumps are sharing a common load. If two pumps are all in new state (state 0), there is no interaction between them (i.e., their transition probabilities remain unchanged) because they operate with their nominal load. However, when a pump is in a degraded state (state 1 or 2) leading to its performance reduction or if it is in failed state (state 3) meaning that it stops working, the other pump needs to work harder to realize the same common load. As a consequence, the degradation process of the other pump is accelerated due to the overload induced by the degraded/failed pump. To demonstrate the degradation interaction in the considered pumping system via the proposed model, we set $\alpha^1 = \alpha^2 = 1.0$ and assume that the degradation interaction matrix is given in the following:

$$\boldsymbol{Z} = \begin{bmatrix} 0.0 & 0.4 \\ 0.4 & 0.0 \end{bmatrix} \tag{5.9}$$

Considering specifically the case where pump 1 is as good as new, the transition probabilities at new state of pump 1 change depending on condition states of pump 2 according to the state interaction between two pumps. Figure 5.3 reports the changes in the transition probabilities at new state of pump 1 when pump 2 is in different condition states. It can be noticed that as pump 2 deteriorates, the possibility that pump 1 is still in new state (p_{00}^1) declines, while it is more likely that pump 1 enters to more degrading condition states at the next transition due to the increase of p_{01}^1, p_{02}^1, p_{03}^1.

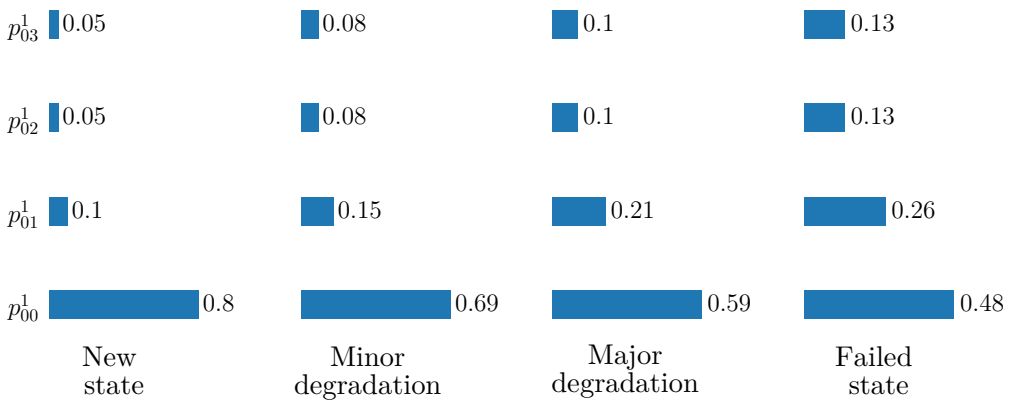


Figure 5.3: Illustration of the changes in transition probabilities at new state of pump 1.

5.3 Maintenance decision optimization

The maintenance approach proposed in Section 4.3 is used to optimize maintenance policies for the series-parallel system described in previous section. Particularly, the degradation interaction model and a maintenance cost model of the studied system (Subsection 5.3.1) are integrated into the framework of MADRL to optimize maintenance decisions. Moreover, different observability settings for cooperative reinforcement learning agents are considered for maintenance planning optimization in Subsection 5.3.2.

5.3.1 Maintenance cost

Cost of separately maintaining a component i of the system studied in this chapter is composed of four elements in which there are two levels of setup cost following [52]:

$$c_k^i = c^{ins,i} + \mathbb{I}_k^{m,c,i}.\left(c^{s,type,j} + c^{s,sys} + c_k^{m,i}\right) \tag{5.10}$$

where:

- $c^{ins,i}$ is the cost induced by inspecting component i which must be paid even for survival components in order to reveal their current health conditions;

- $\mathbb{I}_k^{m,c,i}$ is the maintenance indicator of component i at timestep t_k whose value is equal to one if component i is maintained or equal to zero otherwise;
- $c^{s,sys}$ is the system setup cost caused by, for instance, administrative handling or transportation of spare parts and is assumed to be the same for all components;
- $c^{s,type,j}$ is the component-type setup cost originated from the requirement of repairman skills or specific tools;
- $c^{m,i}$ is the component-specific maintenance cost of component i which depends on maintenance quality and the component's own characteristics. $c^{m,i}$ is computed according to equation (4.5).

From a practical point of view, maintenance cost of a group of several components can be saved via setup cost sharing mechanisms thanks to the positive economic dependence between them [32]. For our studied system, it is assumed that if several maintenance actions are grouped, the system setup cost has to paid only once. Similarly, the component-type setup costs are charged only once if some components of the same type are maintained together. As a result, the maintenance cost at system level denoted as c_k^{sys} is given by:

$$c_k^{sys} = \sum_{i=1}^{N} c_k^i - \mathbb{I}_k^{m,sys}.(N_k^{m,sys} - 1).c^{s,sys} - \sum_{j=1}^{N^{sub}} \mathbb{I}_k^{m,sub,j}.(N_k^{m,sub,j} - 1).c^{s,type,j} \tag{5.11}$$

where:

- $\mathbb{I}_k^{m,sys}$ is the system maintenance indicator at timestep t_k whose value is equal to one if there is at least one component being maintained or equal to zero otherwise;
- $\mathbb{I}_k^{m,sub,j}$ is the maintenance indicator of subsystem j at timestep t_k whose value is equal to one if there is at least one component of the subsystem (type j) being maintained or equal to zero otherwise;
- $N_k^{m,sys} = \sum_{i=1}^{N} \mathbb{I}_k^{m,c,i}$ is the number of maintained components at timestep t_k;
- $N_k^{m,sub,j}$ is the number of maintained components of subsystem j at timestep t_k.

It should be noted that in the case where individual cost models at component level are

difficult to obtain, the maintenance cost model expressed in equation (5.11) can be learned by an ANN whose architecture is depicted in Figure 4.2 from a historical condition maintenance dataset whose structure is presented in Section 4.3.1.2.

5.3.2 MADRL-based maintenance decision optimization

Maintenance planning problem of the studied series-parallel system is considered as a cooperative maintenance task carried by a set $\mathcal{AG}$ of N agents which can either partially or fully observe the system state in their decision making. It is assumed that there are three possible maintenance actions to be chosen at any timestep t_k for an agent i $(i = 1, \ldots, N)$ which are encoded as belows:

$$a_k^i = \begin{cases} 0 & \text{leave component } i \text{ as it is} \\ 1 & \text{implement IM action on component } i \\ 2 & \text{replace component } i \text{ by the one of the same type} \end{cases} \tag{5.12}$$

It should be noted that IM action has random quality on state after maintenance of a component. Specifically, state after maintenance of a component is supposed to be obtained by uniformly discretely sampling from the interval from new state to its state before maintenance [28]. The objective of the agents is to cooperatively learn an optimal long-term maintenance policy that balances the trade-offs between maintenance frequency and unplanned downtime cost.

Dec-POMDPs and MMDPs are employed to model the interaction between multiple agents and the considered series-parallel system in the case where learning agents can partially and fully observe the system state respectively. It is important to note that the definition of state space, reward function of a such Dec-POMDP or MMDP is the same as the ones defined in Section 4.3.2.1, while the action space is restricted to the action space of IM with random quality. Moreover, the probability that a system state after maintenance $(\bar{\boldsymbol{s}}_k)$ at timestep t_k degrades to a system state before maintenance $(\boldsymbol{s}_{k+1})$ at next timestep t_{k+1} is characterized by new transition matrices $\boldsymbol{P}^i$ $(i = 1, \ldots, N)$ presented in Section 5.2 which have interaction

with each other.

Due to the difference between the ability of perceiving the system state of the learning agents in partially and fully observable setting, their local policies are different in these two scenarios as a result. A brief presentation about the form of agent's local polices of these two settings are given as belows:

- In partially observable setting, an agent $AG^i \in \mathcal{AG}$ at timestep t_k receives only an observation $o_k^i = s_k^i$ instead of the true system state $\boldsymbol{s}_k$. Therefore, it has to memorize the history of state-action transitions $h_k^i = [s_0^i, a_0^i, s_1^i, a_1^i, \ldots, a_{k-1}^i, s_k^i]$ based on which the agent conditions a stochastic policy $\pi^i(a_k^i|h_k^i)$. It is worth noting that the CTDE paradigm presented in Section 3.3.3.1 is employed to train decentralized agents in this scenario.
- In fully observable setting, local agents have the ability of accessing the system state ($\boldsymbol{s}_k$) during both training and execution phase. As a result, they make maintenance decisions conditioned on observed states at system level. Therefore, we can denote the maintenance policy of an agent $AG^i \in \mathcal{AG}$ as $\pi^i(a_k^i|\boldsymbol{s}_k)$.

WQMIX algorithm presented in Section 3.3.3 is employed to optimize maintenance policies of the system studied in this chapter in both partially and fully observable setting. The impact of component dependencies and agent's observability on performance and structure of maintenance policies of the studied system optimized by WQMIX are numerically discussed in the next section.

5.4 Numerical experiments

Numerical simulations are first conducted on a small 5-component system in Subsection 5.4.1 to investigate the ability of approaching exact solutions of WQMIX algorithm in both partially and fully observable setting as well as to study the impact of component stochastic dependence on the structural of optimal maintenance policies. The scalability and performance of WQMIX algorithm in partially observable setting is then investigated in Subsection

5.4.2 via simulation experiments carried on a 11-component system.

5.4.1 Five-component system

5.4.1.1 System description

This small system consists of 5 components which are divided into two subsystems of identical components as illustrated in Figure 5.4. Particularly, component 1 and 2 belong to the first subsystem, and the remaining components belongs to the second one.

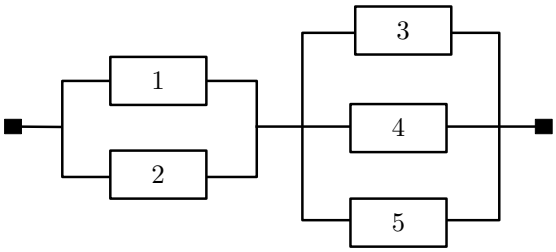


Figure 5.4: Reliability block diagram of the five-component system studied in Chapter 5.

There are 4 health condition states for each component, i.e., *"new state"* (state 0), *"minor damage"* (state 1), *"severe damage"* (state 2) and *"complete failure"* (state 3). The representative inherent transition matrices of the components of the two subsystems are given as follows:

$$\tilde{\boldsymbol{P}}^1 = \begin{bmatrix} 0.80 & 0.10 & 0.05 & 0.05 \\ 0.00 & 0.80 & 0.10 & 0.10 \\ 0.00 & 0.00 & 0.80 & 0.20 \\ 0.00 & 0.00 & 0.00 & 1.00 \end{bmatrix}, \tilde{\boldsymbol{P}}^3 = \begin{bmatrix} 0.70 & 0.20 & 0.05 & 0.05 \\ 0.00 & 0.70 & 0.20 & 0.10 \\ 0.00 & 0.00 & 0.70 & 0.30 \\ 0.00 & 0.00 & 0.00 & 1.00 \end{bmatrix} \tag{5.13}$$

The parameter α^i $(i = 1, 2, ..., 5)$ is set to 1 and it is assumed that the degradation interaction only exists between components of the same subsystem which is described through the following degradation interaction matrix:

$$\boldsymbol{Z} = \begin{bmatrix} 0.0 & 0.4 & 0.0 & 0.0 & 0.0 \\ 0.4 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.2 & 0.2 \\ 0.0 & 0.0 & 0.2 & 0.0 & 0.2 \\ 0.0 & 0.0 & 0.2 & 0.2 & 0.0 \end{bmatrix} \tag{5.14}$$

All cost-related parameters are given in arbitrary units. The system setup cost and the downtime cost are $c^{s,sys} = 20$ and $c^{dt} = 200$ respectively. The component-type setup costs are $c^{s,type,1} = 10$ and $c^{s,type,2} = 5$. The representative maintenance cost parameters of the components of the two subsystems of the studied 5-component system are depicted in Table 5.1.

Table 5.1: Representative cost-related parameters of the 5-component system studied in Chapter 5.

Component i	$c^{ins,i}$	$c^{r,i}$	β^i
1	2	60	5
3	3	40	3

5.4.1.2 Baselines and training descriptions

In this study, WQMIX algorithm is employed to optimize maintenance policies for the five-component system and the results are compared with the ones of Dueling DDQN [103] and VI [99]. In order to facilitate the distinction between WQMIX agents in fully and partially observable setting, we denote the simulation results of these two settings by "F-WQMIX" and "P-WQMIX" respectively. The training descriptions are described in the following paragraphs.

F-WQMIX Centralized reinforcement learning agents are trained through 0.8×10^6 steps with the branching and mixing network whose hyperparameters are presented in Table 5.2 and 5.3. Learning rate is scheduled to decrease from 0.004 to 0.00025 during first 300×10^3

training steps. The constant used for the exploration of learning agents is annealed linearly from 1.0 to 0.05 over 200×10^3 training steps and kept as constant for the rest of the learning. A uniform replay buffer which can store up to 300×10^3 system transitions is employed as a dynamic dataset with a mini-batch of size 128 is uniformly sampled to update agents' network. We set the discount factor $\gamma = 0.99$. The target update frequency is 20×10^3 steps. To monitor learning progress, latest policy networks after every 10^3 steps are employed to generate 5×10^3 consecutive maintenance decisions on a validation environment to calculate corresponding cost rates.

P-WQMIX The local agent networks of P-WQMIX contain a fully-connected layer of 128 hidden units followed by a GRU of 64-dimensional hidden states and a fully-connected layer of 64 hidden units. The configuration of the mixing networks, training step, discount factor, learning rate scheduling and policy evaluation are the same to the ones of F-WQMIX.

While the branching network of F-WQMIX is updated via batches of single system transition point, the local agent networks of P-WQMIX are updated through the gradient of sequences of system transitions due to the incorporation of a recurrent neural network. As a result, a replay buffer that can hold upto 2000 sequences of 100 system transitions is employed. A mini-batch size of 64 sequences is sampled uniformly from this buffer to update the agents' local network.

Dueling DDQN The size of the dueling network's input layer is 5 and its shared MLP is composed of two hidden layers of 256 units. A single fully-connected layer of 128 hidden units is used to compute the system advantage and value function. The training setup for Dueling DDQN is similar to F-WQMIX.

VI This dynamic programming algorithm can be used here to optimize maintenance decisions thanks to the small state and action space of the studied system which are 1024 and 243 respectively. The VI algorithm is trained until the expected estimation accuracy of the value function reaches 0.001. The obtained policy is evaluated via simulation results through

5 runs of 5000 maintenance interventions.

Table 5.2: Hyperparameters of the branching networks used for the simulation experiments conducted in Chapter 5

	5-component system	11-component system
Number of inputs	5	11
Number of outputs in each branch	3	3
Number of neurons in the shared MLP	[128, 128]	[256, 256]
Number of neurons the value MLP	[128]	[128]
Number of neurons the advantage ML	[64]	[64]

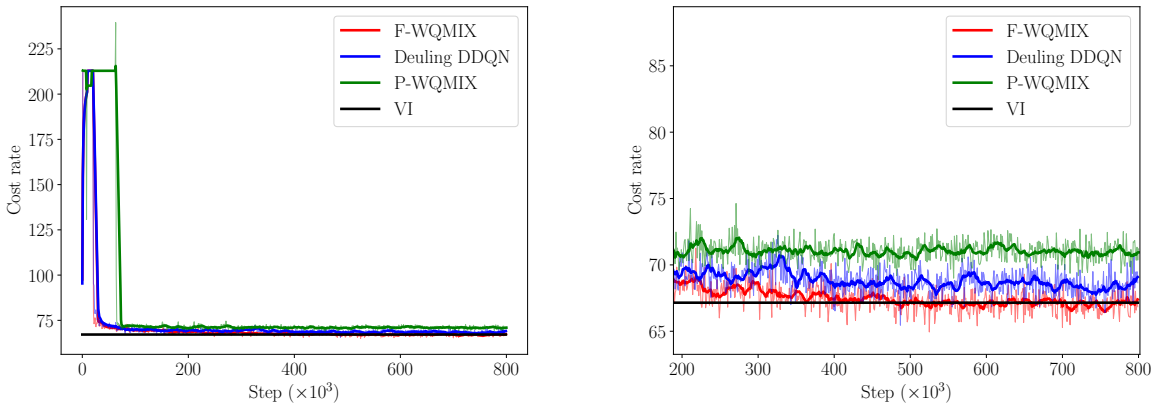
Table 5.3: Hyperparameters of the mixing networks used for the simulation experiments conducted in Chapter 5

	5-component system	11-component system
Number of inputs	5	11
Number of outputs	1	1
Number of neurons in hidden layers of $\boldsymbol{Q}$	[64, 64]	[128, 128]
Number of neurons in hidden layers of $\boldsymbol{Q}^{tot}$	[64, 64]	[128, 128]

5.4.1.3 Simulation results

The evolution of cost rates during training of the three RL algorithms is depicted in Figure 5.5 and their best found cost rates and training time are summarized in the first row of Table 5.4. It can be noticed that P-WQMIX and Dueling DDQN are incapable of approaching the global optimal policy obtained by VI and F-WQMIX with the corresponding cost rate of 67.15. The lower performance of P-WQMIX in comparison to F-WQMIX can be explained through the partial observability that causes difficulties for local agents to make "good" maintenance decisions considering the dependencies between components. For the reason

why Dueling DDQN cannot approach the optimal global policy, we think that it is due to the relatively large size of the action space of the studied system which is 243 that can harm the performance of single-agent DRL algorithms.



(a) The evolution of cost rates during training (b) A closer look at the evolution of cost rates

Figure 5.5: Agent training monitoring for maintenance optimization of the 5-component system studied in Chapter 5.

Based on the second row of Table 5.4, one can see that the computing time of VI is largest that confirms the well-known optimization time problem of dynamic programming algorithms. The training time of F-WQMIX and Dueling DDQN is relatively reasonable for this small system. In comparison to two other reinforcement learning algorithms, P-WQMIX has the largest training time due to the update scheme used for recurrent neural networks.

Table 5.4: Summary of cost rates optimized by different algorithms with corresponding optimization times for the 5-component system studied in Chapter 5.

	F-WQMIX	Dueling DDQN	P-WQMIX	VI
Cost rate	67.15	68.75	71.25	67.15
Optimization time (hour)	2.12	1.58	4.42	9.55

5.4.1.4 Sensitivity analysis

In this sensitivity analysis study, we first investigate the performance of a maintenance policy that is optimized based on a model that does not correctly represent the "ground-truth" system and then investigate the impact of stochastic dependence on the behavior of an optimal maintenance policy.

For these purposes, a similar experiment to the previous one is conducted except that in the maintenance model, the stochastic dependence between components is not included. However, the maintenance policy optimized based this model is employed to make maintenance decisions on the "ground-truth" system where there exists the stochastic dependence between components to compute the corresponding cost rate. The optimization process is carried by using VI or F-WQMIX with the objective of providing exact solutions. The obtained cost rate of the experiment is 76.07 which is larger than the one of the previous experiment (67.15) confirming that omitting component dependencies in maintenance modeling results in the finding of suboptimal maintenance policies.

Table 5.5: Illustration of the optimized maintenance policy with and without considering component stochastic dependence for the 5-component system studied in Chapter 5.

$\boldsymbol{s}_k$	$\boldsymbol{a}_k^*$	$\boldsymbol{a}_k$
[2, 3, 3, 2, 1]	[1, 2, 2, 1, 0]	[0, 2, 2, 0, 0]
[2, 3, 3, 2, 2]	[1, 2, 2, 1, 1]	[0, 2, 2, 0, 0]
[2, 3, 3, 2, 3]	[1, 2, 2, 1, 2]	[0, 2, 2, 1, 0]
[2, 3, 3, 3, 0]	[1, 2, 2, 2, 0]	[0, 2, 2, 0, 0]
[3, 2, 1, 2, 2]	[2, 1, 0, 1, 1]	[2, 0, 0, 0, 0]
[3, 2, 1, 2, 3]	[2, 1, 0, 1, 2]	[2, 0, 0, 0, 2]
[3, 2, 1, 3, 0]	[2, 1, 0, 2, 0]	[2, 0, 0, 0, 0]
[3, 2, 1, 3, 1]	[2, 1, 0, 2, 0]	[2, 0, 0, 0, 0]
[3, 2, 1, 3, 2]	[2, 1, 0, 2, 1]	[2, 0, 0, 2, 0]

Table 5.5: Illustration of the optimized maintenance policy with and without considering component stochastic dependence for the 5-component system studied in Chapter 5.

$\boldsymbol{s}_k$	$\boldsymbol{a}_k^*$	$\boldsymbol{a}_k$
[3, 3, 3, 1, 0]	[2, 2, 2, 0, 0]	[2, 2, 0, 0, 0]
[3, 3, 3, 1, 1]	[2, 2, 2, 0, 0]	[2, 2, 0, 0, 0]
[3, 3, 3, 1, 2]	[2, 2, 2, 0, 1]	[2, 2, 2, 0, 0]
[3, 3, 3, 1, 3]	[2, 2, 2, 0, 2]	[2, 2, 2, 0, 0]
[3, 3, 3, 2, 0]	[2, 2, 2, 1, 0]	[2, 2, 0, 0, 0]
[3, 3, 3, 3, 2]	[2, 2, 2, 2, 1]	[2, 2, 2, 0, 1]
[3, 3, 3, 3, 3]	[2, 2, 2, 2, 2]	[2, 2, 2, 2, 0]

An illustration of the two optimized maintenance policies is depicted in Table 5.5 showing that the optimal policy ($\boldsymbol{a}_k^*$) has an incentive to perform maintenance actions more frequently than the non-optimal one ($\boldsymbol{a}_k$) due to the fact that the stochastic dependence between components makes the system more prone to be shut down if a component degrades to a severe health condition state. Moreover, it can be noticed that IM actions are implemented more regularly for the system with stochastic dependence as a way to improve the system reliability with lower cost than conducting replacement actions.

5.4.2 Eleven-component system

In this section, the scalability and performance of WQMIX algorithm in both fully and partially observable setting are examined via simulation experiments conducted on a 11-component series-parallel system.

5.4.2.1 System description

The examined series-parallel system consists of 11 components which are grouped into five subsystems of identical components as depicted in Figure 5.6. Specifically, component 1 is

considered as the first subsystem. The next subsystem is composed of component 2 and 3. Component 4, 5 and 6 together form the third subsystem. The fourth subsystem contains component 7 and 8. The last one consists of the remaining components.

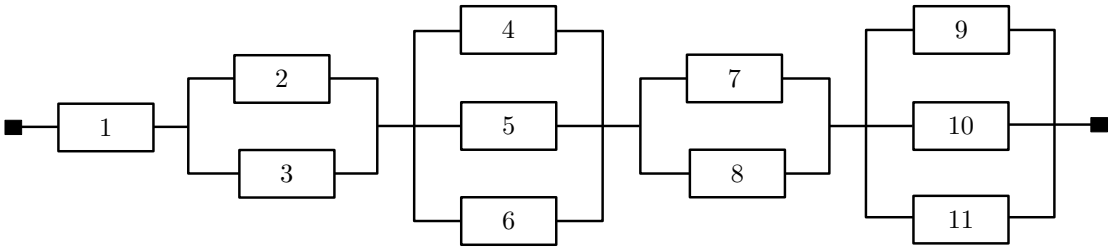


Figure 5.6: Reliability block diagram of the 11-component system studied in Chapter 5.

Each component has 4 condition states using the same definition as used for the five-component system studied in previous section. The components of subsystem 1 and 2 degrade according to two representative transition matrix $\tilde{\boldsymbol{P}}^1$ and $\tilde{\boldsymbol{P}}^3$ expressed in equation (5.13) if there is no stochastic dependence in these subsystems. The representative inherent transition matrices of three remaining subsystems are given as belows:

$$\tilde{\boldsymbol{P}}^6 = \begin{bmatrix} 0.65 & 0.25 & 0.05 & 0.05 \\ 0.00 & 0.65 & 0.25 & 0.10 \\ 0.00 & 0.00 & 0.65 & 0.35 \\ 0.00 & 0.00 & 0.00 & 1.00 \end{bmatrix}, \tilde{\boldsymbol{P}}^8 = \begin{bmatrix} 0.60 & 0.30 & 0.05 & 0.05 \\ 0.00 & 0.60 & 0.30 & 0.10 \\ 0.00 & 0.00 & 0.60 & 0.40 \\ 0.00 & 0.00 & 0.00 & 1.00 \end{bmatrix} \tag{5.15}$$

$$\tilde{\boldsymbol{P}}^{11} = \begin{bmatrix} 0.55 & 0.35 & 0.05 & 0.05 \\ 0.00 & 0.55 & 0.35 & 0.10 \\ 0.00 & 0.00 & 0.55 & 0.45 \\ 0.00 & 0.00 & 0.00 & 1.00 \end{bmatrix} \tag{5.16}$$

In this experiment, we assume that the stochastic dependence can exist between components of different subsystems. Particularly, the degradation interaction exits between the components of subsystem 1 and 2, and between the components of subsystem 4 and 5. Moreover,

the stochastic dependence also exists between components of the same subsystem. The degradation interaction matrix expressed in equation (5.17) is employed to describe component stochastic interactions in this case study.

$$\boldsymbol{Z} = \begin{bmatrix} 0.0 & 0.1 & 0.1 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.1 & 0.0 & 0.1 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.1 & 0.1 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.1 & 0.1 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.1 & 0.0 & 0.1 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.1 & 0.1 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.1 & 0.1 & 0.1 & 0.1 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.1 & 0.0 & 0.1 & 0.1 & 0.1 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.1 & 0.1 & 0.0 & 0.1 & 0.1 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.1 & 0.1 & 0.1 & 0.0 & 0.1 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.1 & 0.1 & 0.1 & 0.1 & 0.0 \end{bmatrix} \tag{5.17}$$

The system setup cost and the downtime cost of the considered system are increased to $c^{s,sys} = 150$ and $c^{dt} = 600$ respectively due to the larger system's size. The component-type setup costs are $c^{s,type,1} = 10$, $c^{s,type,2} = 5$, $c^{s,type,3} = 15$, $c^{s,type,4} = 10$, and $c^{s,type,5} = 5$. The other representative maintenance-related cost parameters of the five subsystems are given in Table 5.6.

Table 5.6: Representative cost-related parameters of the 11-component system studied in Chapter 5.

Component i	$c^{ins,i}$	$c^{r,i}$	β^i
1	2	60	5
3	3	40	3
6	4	35	4
8	5	30	3
11	4	35	3

5.4.2.2 Baselines and training descriptions

VI and Dueling DDQN are not applicable here due to the high dimensionality of state and action space of the studied system. Particularly, the total number of system states is greater than 4×10^6 and the number of all possible system actions is more than 177×10^3. Therefore, a conventional threshold-based maintenance policy are employed to evaluate the performance of F-WQMIX and P-WQMIX. The training configuration of the two MADRL algorithms and threshold optimization process via GA optimizer are given in the following paragraphs.

F-WQMIX F-WQMIX algorithm is trained through 3×10^6 steps with a larger branching and mixing network whose hyperparameters are presented in Table 5.2 and 5.3.Learning rate is scheduled to decrease linearly from 0.001 to 0.00025 during the first 500×10^3 training steps. The mini-batch size, discount factor and the size of replay buffer are 128, 0.99, and 500×10^3 respectively. The target update frequency is 20×10^3 steps.

P-WQMIX The configuration used for training the local agent networks of P-WQMIX in this study is the same as the one used in previous experiment except that the replay buffer now can hold up to 3000 sequences of 300 system transitions.

Threshold-based maintenance policy This classical maintenance policy originates from [28] which can be expressed by a vector $\boldsymbol{l} = \left[l^1, l^2, \ldots, l^N\right]$ where l^i is the preventive maintenance threshold of component i. The detail description of maintenance schedule of a component i is given in the following:

- If $s_k^i = m^i$, the component is in failed condition state. Therefore, the "replacement" action is carried out immediately.
- If $l^i \leq s_k^i < m^i$, the component is still functioning but badly. Thus, the "imperfect maintenance" action is implemented.
- If $s_k^i < l^i$, the component is functioning well. Hence, the "do nothing" action is chosen.

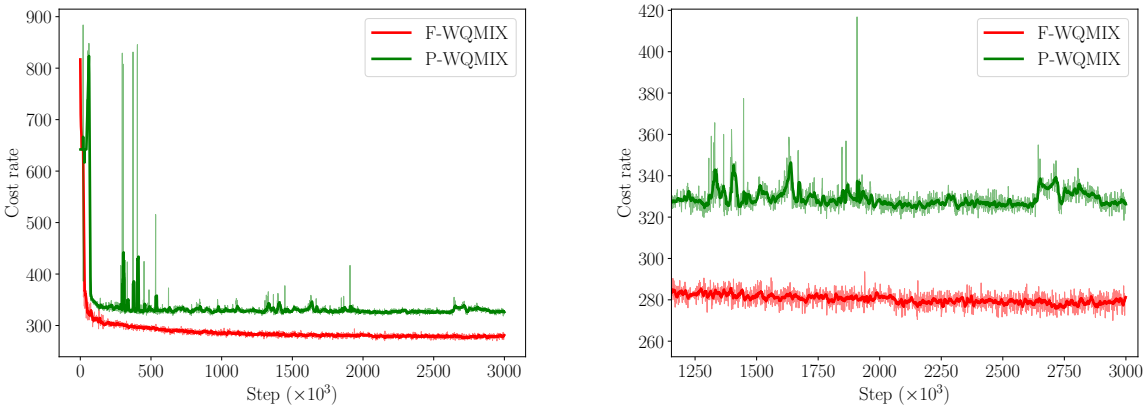
The optimal system-level preventive maintenance thresholds are obtained via GA optimizer.

The GA algorithm is initialized with the population's size of 20 and mutation rate of 0.1. During the optimization process, each potential solution is evaluated using simulation results from 5 runs of 10^4 maintenance interventions. The training converges after 25 iterations.

5.4.2.3 Simulation results

The simulation results are presented in Figure 5.7 and Table 5.7. It can be noticed that the optimized cost rate of the studied threshold-based policy is the highest which is 347.68 with corresponding preventive maintenance threshold: [1 2 2 2 2 2 2 2 2 2 2]. The F-WQMIX agents found the best cost rate of 279.85.

Moreover, it can also be noticed that the performance of P-WQMIX might be harmed if the size of maintained systems gets larger. In particular, the optimized cost rates of F-WQMIX are 6.11% (71.25/61.15 − 1) and 17.41% (328.56/279.85 − 1) greater than the ones obtained by P-WQMIX for the studied 5-component and 11-component system respectively.



(a) The evolution of cost rates during training. (b) A closer look at the evolution of cost rates.

Figure 5.7: Agent training monitoring for maintenance optimization of the 11-component system studied in Chapter 5.

Based on Table 5.7, we can see that the amount of time needed to optimize the threshold-based policy is shortest because the search space of preventive maintenance thresholds of the studied system is not too large. It can be also noticed that the training time of P-WQMIX

explodes due to the larger number of training steps as well as larger length of the sequence of system transitions used to train agent networks.

Table 5.7: Summary of cost rates optimized by different algorithms with corresponding optimization times for the 11-component system studied in Chapter 5.

	F-WQMIX	P-WQMIX	Threshold-based policy
Cost rate	279.85	328.56	347.68
Optimization time (hour)	26.33	107.25	1.55

5.5 Conclusions

In this chapter, a novel state-rate degradation interaction model is proposed to describe the stochastic dependence in a system consisting of discrete-state components. In particular, the proposed model allows modeling the physical phenomenon that the degradation of a component can accelerate the deterioration speed of other working components. Additionally, the system is also supposed to have a parallel structure which allows to characterize the component economic dependence via a multiple setup-cost sharing model.

The maintenance approach proposed in Chapter 4 is applied to optimize maintenance decisions of the systems studied in this chapter in both fully and partially observable setting. Comprehensive numerical studies are conducted to investigate the impact of component dependencies and agent's observability on the performance and structure of optimized maintenance polices. The simulation results show that optimized maintenance policies have an intensive to implement maintenance actions more frequently and in group due to the impact of component dependencies, and the performance of WQMIX algorithm might be harmed in partially observable setting when the size of maintained systems gets larger.

Chapter 6

Conclusions and perspectives

In this thesis, we considered the problem of finding a scalable optimization approach for CBM decision-making of multi-component systems. Based on the literature review carried out in Chapter 2, MADRL is identified as a promising framework for optimizing maintenance policies for systems composed of multiple dependent components because it allows to deal with sequential decision-making problems under uncertainty with both large state and action spaces. MADRL-based maintenance planning for multi-component systems usually requires the construction of maintenance cost models at system level characterizing the economic dependence between maintained components which is used in the formulation of the reward function, and degradation dependence models characterizing the component stochastic dependence which is employed to describe the probabilistic transition dynamic. These requirements lead to the two identified scientific issues as well as the two corresponding contributions established in this PhD.

Particularly, a maintenance planning optimization approach is proposed in Chapter 4 for multi-component systems, which integrates the degradation and maintenance cost model of the system under consideration into the framework of MADRL, allowing to deal with the curse of dimensionality that optimization algorithms must face when they are used to optimize maintenance decisions of large systems. Moreover, maintenance cost models at system level are supposed to be learned by artificial neural networks from dedicated condition monitoring data that allows to get rid of the traditional demand of accessing individual costs at component level to compute system maintenance costs. The proposed maintenance approach is useful for real industrial applications because accessing individual costs becomes very problematic in practice. Indeed, maintenance actions are often grouped in each maintenance intervention due to component economic dependence, which leads to the fact that individual costs, such as setup costs, spare part costs, maintenance labor costs, etc. are not

recorded separately, instead, only total cost is documented.

The second contribution of this PhD is the proposal of a novel state-rate degradation interaction model based on Markov processes, which means that the degradation of a component can accelerate the deterioration speed of other components. This contribution fill the gap in modeling the stochastic dependence via degradation interactions for systems consisting of multi-state components which degrade discretely over time. The proposed degradation dependence model can be easily integrated into the framework of MADRL along with the proposed deep maintenance cost model at system level learned from condition monitoring data to optimize maintenance decisions for multi-component systems.

Although the contributions are quite generic and can be applied to different types of multi-component systems in reality, there still exits some limitations that can be considered for improvements as listed below:

- Firstly, the maintenance cost models at system level considered in this PhD is assumed to be dependent only on the state before and after maintenance of the components, and then can be learned by feedforward neural networks. This kind of deep model might not be suitable for some situations where system maintenance costs are time-dependent. In such cases, different kinds of recurrent neural networks such as long short-term memory and transformer network might be potential solutions.

- Secondly, the proposed degradation dependence model is quite general and not restricted by the system structure from the reliability block-diagram point of view. However, the systems studied in Chapter 5 is assumed to have a series-parallel structure due to the employment of a multi-setup sharing model characterizing the component economic dependence. Therefore, multi-component systems with more complex structures should be taken into consideration to investigate the impact of stochastic dependence descried by the proposed model on optimal maintenance policies' structure.

- Finally, although the WQMIX algorithm used in this PhD showed good performance in optimizing maintenance policies for multi-component systems, its computational time

seems not so reasonable. One reason is that WQMIX belongs to the class of model-free RL algorithms which learn optimal sequential policies by purely interacting with their environment without any prior knowledge. Hence, using expert knowledge to reduce the size of policy search space for MADRL algorithms could be an interesting research direction for maintenance decision-making and optimization.

Appendix A

Cost-related parameters of the systems studied in chapter 4

A.1 Four-component system

The setup cost and downtime cost are $c^{s,sys} = 50$ and $c^{dt} = 500$. The other parameters related to imperfect maintenance are given in Table A.1.

Table A.1: IM parameters of the 4-component system studied in Chapter 4.

i	1	2	3	4
$c^{r,i}$	70	65	50	55
$c^{ins,i}$	5	2	3	4
β^i	3	2	1	4

A.2 Fifteen-component system

The setup cost and downtime cost are $c^{s,sys} = 200$ and $c^{dt} = 1000$. The components of subsystem $j \in \{1, 2, 3, 4\}$ are maintained following the parameters $c^{r,j}, c^{ins,j}, \beta^j$ which are presented in Table A.1. The cost-related parameters of the last subsystem are $c^{r,5} = 45$, $c^{ins,5} = 3$, $\beta^5 = 5$.

Publications of the author

Journal papers:

- Van-Thai Nguyen, Phuc Do, Alexendre Voisin and Benoit Iung (2022). *"Artificial-intelligence-based maintenance decision-making and optimization for multi-state component systems"*. Reliability Engineering & System Safety, 228, 108757.

Conference papers:

- Van-Thai Nguyen, Phuc Do, Alexendre Voisin and Benoit Iung (2022). *"Weighted-QMIX-based optimization for maintenance decision-making of multi-component systems"*. In PHM Society European Conference (Vol. 7, No. 1, pp. 360-367).
- Van-Thai Nguyen, Phuc Do, Alexandre Voisin and Benoit Iung (2021). *"Reinforcement learning for maintenance decision-making of multi-state component systems with imperfect maintenance"*. Proceedings of the 31st European Safety and Reliability Conference.

References

1. De Jonge, B. & Scarf, P. A. A review on maintenance optimization. *European journal of operational research* **285,** 805–824 (2020).
2. Shin, J.-H. & Jun, H.-B. On condition based maintenance policy. *Journal of Computational Design and Engineering* **2,** 119–127 (2015).
3. Zhang, N. & Si, W. Deep reinforcement learning for condition-based maintenance planning of multi-component systems under dependent competing risks. *Reliability Engineering & System Safety* **203,** 107094 (2020).
4. Savolainen, J. & Urbani, M. Maintenance optimization for a multi-unit system with digital twin simulation. *Journal of Intelligent Manufacturing* **32,** 1953–1973 (2021).
5. Thomas, D. S. *The costs and benefits of advanced maintenance in manufacturing* (US Department of Commerce, National Institute of Standards and Technology ..., 2018).
6. Vachon, W. *Long-term O&M costs of wind turbines based on failure rates and repair costs* in *Proceedings WINDPOWER, American Wind Energy Association annual conference, Portland, OR* (2002), 2–5.
7. Waeyenbergh, G. & Pintelon, L. A framework for maintenance concept development. *International journal of production economics* **77,** 299–313 (2002).
8. Bowman, R. A. & Schmee, J. Pricing and managing a maintenance contract for a fleet of aircraft engines. *Simulation* **76,** 69–77 (2001).
9. Rausand, M. & Hoyland, A. *System reliability theory: models, statistical methods, and applications* (John Wiley & Sons, 2003).
10. Ahmad, R. & Kamaruddin, S. An overview of time-based and condition-based maintenance in industrial application. *Computers & industrial engineering* **63,** 135–149 (2012).
11. Huang, J., Chang, Q. & Arinez, J. Deep reinforcement learning based preventive maintenance policy for serial production lines. *Expert Systems with Applications* **160,** 113701 (2020).

12. Dalzochio, J., Kunst, R., Pignaton, E., Binotto, A., Sanyal, S., Favilla, J. & Barbosa, J. Machine learning and reasoning for predictive maintenance in Industry 4.0: Current status and challenges. *Computers in Industry* **123,** 103298 (2020).
13. Roda, I. & Macchi, M. Maintenance concepts evolution: a comparative review towards Advanced Maintenance conceptualization. *Computers in Industry* **133,** 103531 (2021).
14. Alaswad, S. & Xiang, Y. A review on condition-based maintenance optimization models for stochastically deteriorating system. *Reliability Engineering & System Safety* **157,** 54–63 (2017).
15. Wang, H. A survey of maintenance policies of deteriorating systems. *European journal of operational research* **139,** 469–489 (2002).
16. Nicolai, R. P. & Dekker, R. Optimal maintenance of multi-component systems: a review. *Complex system maintenance handbook,* 263–286 (2008).
17. Su, J., Huang, J., Adams, S., Chang, Q. & Beling, P. A. Deep multi-agent reinforcement learning for multi-level preventive maintenance in manufacturing systems. *Expert systems with applications* **192.** ISSN: 0957-4174 (2022).
18. Andriotis, C. & Papakonstantinou, K. Managing engineering systems with large state and action spaces through deep reinforcement learning. *Reliability Engineering & System Safety* **191,** 106483 (2019).
19. Andriotis, C. & Papakonstantinou, K. Deep reinforcement learning driven inspection and maintenance planning under incomplete information and constraints. *Reliability Engineering & System Safety* **212,** 107551 (2021).
20. Do, P., Assaf, R., Scarf, P. & Iung, B. Modelling and application of condition-based maintenance for a two-component system with stochastic and economic dependencies. *Reliability Engineering & System Safety* **182,** 86–97 (2019).
21. Wildeman, R. E., Dekker, R. & Smit, A. A dynamic policy for grouping maintenance activities. *European Journal of Operational Research* **99,** 530–551 (1997).
22. Do Van, P., Barros, A., Bérenguer, C., Bouvard, K. & Brissaud, F. Dynamic grouping maintenance with time limited opportunities. *Reliability Engineering & System Safety* **120,** 51–59 (2013).

23. Keizer, M. C. O., Flapper, S. D. P. & Teunter, R. H. Condition-based maintenance policies for systems with multiple dependent components: A review. *European Journal of Operational Research* **261,** 405–420 (2017).

24. Wang, Y., Li, X., Chen, J. & Liu, Y. A condition-based maintenance policy for multi-component systems subject to stochastic and economic dependencies. *Reliability Engineering & System Safety* **219,** 108174 (2022).

25. Nguyen, K. T., Do, P., Huynh, K. T., Bérenguer, C. & Grall, A. Joint optimization of monitoring quality and replacement decisions in condition-based maintenance. *Reliability Engineering & System Safety* **189,** 177–195 (2019).

26. Rashid, T., Farquhar, G., Peng, B. & Whiteson, S. Weighted qmix: Expanding monotonic value function factorisation for deep multi-agent reinforcement learning. *Advances in Neural Information Processing Systems* (2020).

27. Do, P., Voisin, A., Levrat, E. & Iung, B. A proactive condition-based maintenance strategy with both perfect and imperfect maintenance actions. *Reliability Engineering & System Safety* **133,** 22–32 (2015).

28. Do, V. P. & Bérenguer, C. Condition-based maintenance with imperfect preventive repairs for a deteriorating production system. *Quality and Reliability Engineering International* **28,** 624–633 (2012).

29. Yousefi, N., Tsianikas, S. & Coit, D. W. Reinforcement learning for dynamic condition-based maintenance of a system with individually repairable components. *Quality Engineering* **32,** 388–408 (2020).

30. Quatrini, E., Costantino, F., Di Gravio, G. & Patriarca, R. Condition-based maintenance—an extensive literature review. *Machines* **8,** 31 (2020).

31. Thomas, L. A survey of maintenance and replacement models for maintainability and reliability of multi-item systems. *Reliability Engineering* **16,** 297–309 (1986).

32. Dekker, R., Wildeman, R. E. & Van der Duyn Schouten, F. A. A review of multi-component maintenance models with economic dependence. *Mathematical methods of operations research* **45,** 411–435 (1997).

33. Goyal, D. & Pabla, B. Condition based maintenance of machine tools—A review. *CIRP Journal of Manufacturing Science and Technology* **10,** 24–35 (2015).

34. Dao, C. D. & Zuo, M. J. Selective maintenance of multi-state systems with structural dependence. *Reliability engineering & system safety* **159,** 184–195 (2017).

35. Dinh, D.-H., Do, P. & Iung, B. Degradation modeling and reliability assessment for a multi-component system with structural dependence. *Computers & Industrial Engineering* **144,** 106443 (2020).

36. Tsang, A. H. Condition-based maintenance: tools and decision making. *Journal of quality in maintenance engineering* (1995).

37. Ahmad, R. & Kamaruddin, S. A review of condition-based maintenance decision-making. *European journal of industrial engineering* **6,** 519–541 (2012).

38. Guillén, A. J., Crespo, A., Gómez, J. F. & Sanz, M. D. A framework for effective management of condition based maintenance programs in the context of industrial development of E-Maintenance strategies. *Computers in Industry* **82,** 170–185 (2016).

39. Zonta, T., Da Costa, C. A., da Rosa Righi, R., de Lima, M. J., da Trindade, E. S. & Li, G. P. Predictive maintenance in the Industry 4.0: A systematic literature review. *Computers & Industrial Engineering* **150,** 106889 (2020).

40. Van Noortwijk, J. M. A survey of the application of gamma processes in maintenance. *Reliability Engineering & System Safety* **94,** 2–21 (2009).

41. Papakonstantinou, K. & Shinozuka, M. Optimum inspection and maintenance policies for corroded structures using partially observable Markov decision processes and stochastic, physically based models. *Probabilistic Engineering Mechanics* **37,** 93–108 (2014).

42. Papakonstantinou, K. G. & Shinozuka, M. Planning structural inspection and maintenance policies via dynamic programming and Markov processes. Part I: Theory. *Reliability Engineering & System Safety* **130,** 202–213 (2014).

43. Papakonstantinou, K. G. & Shinozuka, M. Planning structural inspection and maintenance policies via dynamic programming and Markov processes. Part II: POMDP implementation. *Reliability Engineering & System Safety* **130,** 214–224 (2014).

44. Shi, H. & Zeng, J. Real-time prediction of remaining useful life and preventive opportunistic maintenance strategy for multi-component systems considering stochastic dependence. *Computers & Industrial Engineering* **93,** 192–204 (2016).

45. Dong, W., Liu, S. & Du, Y. Optimal periodic maintenance policies for a parallel redundant system with component dependencies. *Computers & Industrial Engineering* **138,** 106133 (2019).

46. Dijkhuizen, G., Rahim, M. & Ben-Daya, M. Maintenance grouping in multi-setup multi-component production systems. *Integrated Models in Production Planning, Inventory, Quality and Maintenance, Boston: Kluwer Academic Publishers,* 283–306 (2001).

47. Vu, H. C., Do, P. & Barros, A. A stationary grouping maintenance strategy using mean residual life and the birnbaum importance measure for complex structures. *IEEE Transactions on reliability* **65,** 217–234 (2015).

48. Dinh, D.-H., Do, P. & Iung, B. Multi-level opportunistic predictive maintenance for multi-component systems with economic dependence and assembly/disassembly impacts. *Reliability Engineering & System Safety* **217,** 108055 (2022).

49. Derman, C. On optimal replacement rules when changes of state are Markovian. *Mathematical optimization techniques* **396,** 201–210 (1963).

50. Kurt, M. & Kharoufeh, J. P. Monotone optimal replacement policies for a Markovian deteriorating system in a controllable environment. *Operations Research Letters* **38,** 273–279 (2010).

51. Kim, D.-Y., Kim, S., Hassan, H. & Park, J. H. Adaptive data rate control in low power wide area networks for long range IoT services. *Journal of computational science* **22,** 171–178 (2017).

52. Wijnmalen, D. J. & Hontelez, J. A. Coordinated condition-based repair strategies for components of a multi-component maintenance system with discounts. *European Journal of Operational Research* **98,** 52–63 (1997).

53. Qian, X. & Wu, Y. Condition based maintenance optimization for the hydro generating unit with dynamic economic dependence. *International Journal of Control and Automation* **7,** 317–326 (2014).

54. Satow, T. & Osaki, S. Optimal replacement policies for a two-unit system with shock damage interaction. *Computers & Mathematics with Applications* **46,** 1129–1138 (2003).

55. Sheu, S.-H., Liu, T.-H., Zhang, Z. G. & Ke, J.-C. Extended preventive replacement policy for a two-unit system subject to damage shocks. *International Journal of Production Research* **53,** 4614–4628 (2015).

56. Van Horenbeek, A. & Pintelon, L. A dynamic predictive maintenance policy for complex multi-component systems. *Reliability engineering & system safety* **120,** 39–50 (2013).

57. Marseguerra, M., Zio, E. & Podofillini, L. Condition-based maintenance optimization by means of genetic algorithms and Monte Carlo simulation. *Reliability Engineering & System Safety* **77,** 151–165 (2002).

58. Zhang, Z., Wu, S., Lee, S. & Ni, J. Modified iterative aggregation procedure for maintenance optimisation of multi-component systems with failure interaction. *International Journal of Systems Science* **45,** 2480–2489 (2014).

59. Zhang, Z., Wu, S., Li, B. & Lee, S. (n, N) type maintenance policy for multi-component systems with failure interactions. *International Journal of Systems Science* **46,** 1051–1064 (2015).

60. Keizer, M. C. O., Teunter, R. H., Veldman, J. & Babai, M. Z. Condition-based maintenance for systems with economic dependence and load sharing. *International Journal of Production Economics* **195,** 319–327 (2018).

61. Feng, Q., Jiang, L. & Coit, D. W. Reliability analysis and condition-based maintenance of systems with dependent degrading components based on thermodynamic physics-of-failure. *The International Journal of Advanced Manufacturing Technology* **86,** 913–923 (2016).

62. Feng, Q., Rafiee, K., Keedy, E., Arab, A., Coit, D. W. & Song, S. Reliability and condition-based maintenance for multi-stent systems with stochastic-dependent competing risk processes. *The International Journal of Advanced Manufacturing Technology* **80,** 2027–2040 (2015).

63. Mercier, S. & Pham, H. H. A preventive maintenance policy for a continuously monitored system with correlated wear indicators. *European Journal of Operational Research* **222,** 263–272 (2012).

64. Mercier, S. & Pham, H. H. A condition-based imperfect replacement policy for a periodically inspected system with two dependent wear indicators. *Applied Stochastic Models in Business and Industry* **30,** 766–782 (2014).

65. Fang, G. & Pan, R. On multivariate copula modeling of dependent degradation processes. *Computers & Industrial Engineering* **159,** 107450 (2021).

66. Hong, H.-P., Zhou, W., Zhang, S. & Ye, W. Optimal condition-based maintenance decisions for systems with dependent stochastic degradation of components. *Reliability Engineering & System Safety* **121,** 276–288 (2014).

67. Li, H., Deloux, E. & Dieulle, L. A condition-based maintenance policy for multi-component systems with Lévy copulas dependence. *Reliability Engineering & System Safety* **149,** 44–55 (2016).

68. Rasmekomen, N. & Parlikad, A. K. Optimising maintenance of multi-component systems with degradation interactions. *IFAC Proceedings Volumes* **47,** 7098–7103 (2014).

69. Rasmekomen, N. & Parlikad, A. K. Condition-based maintenance of multi-component systems with degradation state-rate interactions. *Reliability Engineering & System Safety* **148,** 1–10 (2016).

70. Bian, L. & Gebraeel, N. Stochastic framework for partially degradation systems with continuous component degradation-rate-interactions. *Naval Research Logistics (NRL)* **61,** 286–303 (2014).

71. Barbera, F., Schneider, H. & Watson, E. A condition based maintenance model for a two-unit series system. *European Journal of Operational Research* **116,** 281–290 (1999).

72. Castanier, B., Grall, A. & Bérenguer, C. A condition-based maintenance policy with non-periodic inspections for a two-unit series system. *Reliability Engineering & System Safety* **87,** 109–120 (2005).

73. Wang, L., Zheng, E., Li, Y., Wang, B. & Wu, J. *Maintenance optimization of generating equipment based on a condition-based maintenance policy for multi-unit systems* in *2009 Chinese Control and Decision Conference* (2009), 2440–2445.

74. Zhang, Z., Wu, S. & Li, B. *A condition-based and opportunistic maintenance model for a two-unit deteriorating system* in *2011 International Conference on Quality, Reliability, Risk, Maintenance, and Safety Engineering* (2011), 590–595.

75. Xia, T., Xi, L., Zhou, X. & Lee, J. Condition-based maintenance for intelligent monitored series system with independent machine failure modes. *International Journal of Production Research* **51,** 4585–4596 (2013).

76. Koochaki, J., Bokhorst, J. A., Wortmann, H. & Klingenberg, W. The influence of condition-based maintenance on workforce planning and maintenance scheduling. *International Journal of Production Research* **51,** 2339–2351 (2013).

77. Keizer, M. C. O. & Teunter, R. H. *Clustering condition-based maintenance for a multi-unit system with aperiodic inspections* in *Safety and Reliability of Complex Engineered Systems: Proceedings of the 25th European Safety and Reliability Conference, ESREL 2015, Zürich, Switzerland, 7-10 September 2015* (2015), 983–991.

78. Zhou, Y., Zhang, Z., Lin, T. R. & Ma, L. Maintenance optimisation of a multi-state series–parallel system considering economic dependence and state-dependent inspection intervals. *Reliability Engineering & System Safety* **111,** 248–259 (2013).

79. Zhou, Y., Lin, T. R., Sun, Y., Bian, Y. & Ma, L. An effective approach to reducing strategy space for maintenance optimisation of multistate series–parallel systems. *Reliability Engineering & System Safety* **138,** 40–53 (2015).

80. Huynh, K. T., Barros, A. & Bérenguer, C. A reliability-based opportunistic predictive maintenance model for k-out-of-n deteriorating systems. *Chemical engineering transactions* **33,** 493–498 (2013).

81. Huynh, K. T., Barros, A. & Bérenguer, C. Multi-level decision-making for the predictive maintenance of k-out-of-n: F deteriorating systems. *IEEE transactions on Reliability* **64,** 94–117 (2014).

82. Keizer, M. C. O., Teunter, R. H. & Veldman, J. Clustering condition-based maintenance for systems with redundancy and economic dependencies. *European Journal of Operational Research* **251,** 531–540 (2016).

83. Zhang, X. & Zeng, J. Deterioration state space partitioning method for opportunistic maintenance modelling of identical multi-unit systems. *International Journal of Production Research* **53,** 2100–2118 (2015).

84. Sun, Q., Ye, Z.-S. & Chen, N. Optimal inspection and replacement policies for multi-unit systems subject to degradation. *IEEE Transactions on Reliability* **67,** 401–413 (2017).

85. Andersen, J. F., Andersen, A. R., Kulahci, M. & Nielsen, B. F. A numerical study of Markov decision process algorithms for multi-component replacement problems. *European Journal of Operational Research* **299,** 898–909 (2022).

86. Skordilis, E. & Moghaddass, R. A deep reinforcement learning approach for real-time sensor-driven decision making and predictive analytics. *Computers & Industrial Engineering* **147,** 106600 (2020).

87. Zhang, P., Zhu, X. & Xie, M. A model-based reinforcement learning approach for maintenance optimization of degrading systems in a large state space. *Computers & Industrial Engineering* **161,** 107622 (2021).

88. Rocchetta, R., Bellani, L., Compare, M., Zio, E. & Patelli, E. A reinforcement learning framework for optimal operation and maintenance of power grids. *Applied energy* **241,** 291–301 (2019).

89. Pinciroli, L., Baraldi, P., Ballabio, G., Compare, M. & Zio, E. Optimization of the Operation and Maintenance of renewable energy systems by Deep Reinforcement Learning. *Renewable Energy* (2021).

90. Wei, S., Bao, Y. & Li, H. Optimal policy for structure maintenance: A deep reinforcement learning framework. *Structural Safety* **83,** 101906 (2019).

91. Morato, P. G., Andriotis, C. P., Papakonstantinou, K. G. & Rigo, P. Inference and dynamic decision-making for deteriorating systems with probabilistic dependencies

through Bayesian networks and deep reinforcement learning. *Reliability Engineering & System Safety,* 109144 (2023).

92. Su, J., Adams, S. & Beling, P. A. Value-Decomposition Multi-Agent Actor-Critics. *arXiv preprint arXiv:2007.12306* (2020).
93. Yan, Q., Wu, W. & Wang, H. Deep reinforcement learning for distributed flow shop scheduling with flexible maintenance. *Machines* **10,** 210 (2022).
94. Yousefi, N., Tsianikas, S. & Coit, D. W. Dynamic maintenance model for a repairable multi-component system using deep reinforcement learning. *Quality Engineering* **34,** 16–35 (2022).
95. Mohammadi, R. & He, Q. A deep reinforcement learning approach for rail renewal and maintenance planning. *Reliability Engineering & System Safety* **225,** 108615 (2022).
96. Najafi, S. & Lee, C.-G. A Deep Reinforcement Learning Approach for Repair-Based Maintenance of Multi-Unit Systems using Proportional Hazards Model. *Reliability Engineering & System Safety,* 109179 (2023).
97. Watkins, C. J. & Dayan, P. Q-learning. *Machine learning* **8,** 279–292 (1992).
98. Hasselt, H. Double Q-learning. *Advances in neural information processing systems* **23,** 2613–2621 (2010).
99. Sutton, R. S. & Barto, A. G. *Reinforcement learning: An introduction* (MIT press, 2018).
100. Mnih, V. *et al.* Human-level control through deep reinforcement learning. *nature* **518,** 529–533 (2015).
101. Schaul, T., Quan, J., Antonoglou, I. & Silver, D. Prioritized experience replay. *ICLR 2016* (2015).
102. Van Hasselt, H., Guez, A. & Silver, D. *Deep reinforcement learning with double q-learning* in *Proceedings of the AAAI Conference on Artificial Intelligence* **30** (2016).
103. Wang, Z., Schaul, T., Hessel, M., Hasselt, H., Lanctot, M. & Freitas, N. *Dueling network architectures for deep reinforcement learning* in *International conference on machine learning* (2016), 1995–2003.

104. Oliehoek, F. A. & Amato, C. *A concise introduction to decentralized POMDPs* (Springer, 2016).

105. Boutilier, C. *Planning, learning and coordination in multiagent decision processes* in *TARK* **96** (1996), 195–210.

106. Rashid, T., Samvelyan, M., Schroeder, C., Farquhar, G., Foerster, J. & Whiteson, S. *Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning* in *International Conference on Machine Learning* (2018), 4295–4304.

107. Ha, D., Dai, A. & Le, Q. V. Hypernetworks. *arXiv preprint arXiv:1609.09106* (2016).

108. Tavakoli, A., Pardo, F. & Kormushev, P. *Action branching architectures for deep reinforcement learning* in *Proceedings of the AAAI Conference on Artificial Intelligence* **32** (2018).

109. Pham, H. & Wang, H. Imperfect maintenance. *European journal of operational research* **94,** 425–438 (1996).

110. Golabi K Thompson P, H. W. Pontis Technical Manual Prepared forFederal Highway Administration (1992).

111. DeStefano, P. D. & Grivas, D. A. Method for estimating transition probability in bridge deterioration models. *Journal of infrastructure systems* **4,** 56–62 (1998).

112. Mishalani, R. G. & Madanat, S. M. Computation of infrastructure transition probabilities using stochastic duration models. *Journal of Infrastructure systems* **8,** 139–148 (2002).

113. De Jonge, B. Discretizing continuous-time continuous-state deterioration processes, with an application to condition-based maintenance optimization. *Reliability Engineering & System Safety* **188,** 1–5 (2019).

114. Kingma, D. P. & Ba, J. Adam: A method for stochastic optimization. *The 3rd International Conference for Learning Representations* (2014).

115. Oakley, J. L., Wilson, K. J. & Philipson, P. A condition-based maintenance policy for continuously monitored multi-component systems with economic and stochastic dependence. *Reliability Engineering & System Safety,* 108321 (2022).