Contribution à l'optimisation du désassemblage pour la maintenance, la fin de vie et la reconception basée sur l'apprentissage par renforcement

Amal Allagui

THESE DE DOCTORAT

NNT :2023UPAST183

Contribution à l'optimisation du désassemblage pour la maintenance, la fin de vie et la reconception basée sur l'apprentissage par renforcement

Contribution to disassembly optimization for maintenance, end-of-life and redesign based on reinforcement learning

Thèse de doctorat de l'université Paris-Saclay et de l'université de Monastir

École doctorale n°573 : Interfaces : matériaux, systèmes, usages (INTERFACES)
Spécialité de doctorat : Génie Industriel
Graduate School : Sciences de l'ingénierie et des systèmes. Référent : Centrale Supélec

Thèse préparée dans le laboratoire **LGM** de L'ENIM et le laboratoire **Quartz** d'ISAE-Supméca, sous la direction de **Nizar AIFAOUI**, Professeur, la co-direction de **Olivia PENAS**, Ingénieur de Recherche HDR, le co-encadrement de **Moncef HAMMADI**, Maître de conférences - HDR, le co-encadrement de **Régis PLATEAUX**, Maître assistant.

Thèse soutenue à Paris-Saclay, le 01 décembre 2023, par

Amal ALLAGUI

Composition du Jury

Membres du jury avec voix délibérative

Jean-Yves CHOLEY Professeur des Universités, ISAE Supméca	Président
Peggy ZWOLINSKI Professeure des Universités, Grenoble INP	Rapporteure
Abdelfattah MLIKA Professeur des Universités, ENIS (Sousse)	Rapporteur
Frédéric DEMOLY Professeur des Universités, UTBM	Examinateur
Noureddine BEN YAHIA Professeur des Universités, ENSIT	Examinateur

université PARIS-SACLAY | ÉCOLE DOCTORALE Interfaces: matériaux, systèmes, usages

Titre : Contribution à l'optimisation du désassemblage pour la maintenance, la fin de vie et la reconception basée sur l'apprentissage par renforcement

Mots clés : Conception assistée par ordinateur, Apprentissage par renforcement, Optimisation des séquences de désassemblage, Maintenance préventive, Désassemblage de fin de vie, MBSE.

Résumé : Le processus de désassemblage des pièces mécaniques représente une étape cruciale dans le cycle de vie d'un produit, et il est reconnu comme un challenge économique important au niveau industriel. En outre, l'un des défis majeurs concerne la génération automatique des séquences optimisées de désassemblage de mécanismes.

Le sujet de cette thèse adresse l'optimisation du processus de planification des séquences de désassemblage dans un contexte de maintenance ou de fin de vie, et la reconception conséquente des assemblages mécaniques. L'objectif est de fournir un support d'aide à la décision dans la phase de maintenance et/ou de désassemblage en fin de vie. En outre, soulignant l'importance du facteur temps dans l'industrie manufacturière et sa contribution élevée dans les coûts de production des produits, cette thèse vise à trouver un outil capable de fournir une séquence de désassemblage optimale garantissant le minimum de temps de mise en œuvre, tout en minimisant également le temps de génération de cette séquence. Les paramètres d'optimisation des séquences de démontage qui ont été considérés sont les suivants : minimiser le changement de direction lors des opérations de démontage, minimiser le changement d'outil, favoriser l'accès aux pièces d'usure lors du démontage pour la maintenance et favoriser le démontage des pièces de petite taille en premier lieu. Ces résultats ont été exploités pour proposer des alternatives de reconception des architectures CAO des assemblages mécaniques afin d'optimiser les séquences de démontage.

Ainsi, après une analyse du besoin conduite avec une démarche MBSE, un état de l'art approfondi a été réalisé pour examiner les travaux effectués dans ce domaine. Au travers de cette revue, nous avons pu identifier les objectifs de recherche qui ont guidé nos objectifs de thèse. Pour mettre en œuvre ces approches, nous avons développé un outil informatique, cet outil a été réalisé en utilisant les données issues de l'environnement de CAO et d'une plateforme Python. Nous avons validé l'efficacité de cet outil en l'appliquant à plusieurs exemples et en le comparant avec des travaux issus de la littérature.

Title: Contribution to disassembly optimization for maintenance, end-of-life and redesign based on reinforcement learning

Keywords: Computer-aided design, reinforcement learning, optimization of disassembly sequences, preventive maintenance, end-of-life disassembly, MBSE.

Abstract: The process of disassembling mechanical parts represents a crucial stage in the product life cycle, and is recognized as a major economic challenge at industrial level. Furthermore, one of the major challenges is the automatic generation of optimized mechanism disassembly sequences.

The subject of this thesis addresses the optimization of the disassembly sequence planning process in a maintenance or end-of-life context, and the consequent redesign of mechanical assemblies. The aim is to provide decision support in the maintenance and/or disassembly phase at end-of-life. Furthermore, underlining the importance of the time factor in the manufacturing industry and its high contribution to product production costs, this thesis aims to find a tool capable of providing an optimal disassembly sequence guaranteeing the minimum implementation time, while also minimizing the generation time of this sequence. The disassembly sequence optimization parameters considered are as follows: minimizing change of direction during disassembly operations, minimizing tool change, facilitating access to wear parts during disassembly for maintenance purposes, and favoring disassembly of small parts first. These results were used to propose alternative ways of redesigning CAD architectures for mechanical assemblies in order to optimize disassembly sequences.

Following a needs analysis conducted using an MBSE approach, an in-depth state-of-the-art review was carried out to examine the work carried out in this field. Through this review, we were able to identify the research goals that guided our thesis objectives. To implement these approaches, we developed a software tool, using data from the CAD environment and a Python platform. We validated the effectiveness of this tool by applying it to several examples and comparing it with works from the literature.

SOMMAIRE

LISTE DES FIGURES

LISTE DES TABLEAUX

ABREVIATIONS ET NOTATIONS

PSD : Planification de séquence de désassemblage

SD : Séquence de désassemblage

DFSA/D: Design for selectif assembly/disassembly

AG : Algorithme Génétique

ACO : L'Algorithme des Colonies de Fourmis

PSO : Particle Swarm Optimization

RL : Reinforcement learning

EoL : END of Life

MBSE : Model-Based Systems Engineering

SysML : System Modeling Language

bdd : diagrammes de définition de bloc

CAO : conception assistée par ordinateur

PD : plan de désassemblage

PA : plan d'assemblage

DM : De-Manufacturing

HALG : Hierarchical Attributed Liaison Graph

HDGM : Hybrid Disassembly Graph Model

PN: Petri Net

PND : Petri Net de désassemblage

DDL : Degré de liberté

MSP : Maintenance sequence planning

Qi : Index de qualité

Ti : index de temps

AI : intelligence artificielle

RND : Réseaux de Nœuds de Désassemblage

DLBP : Dismantling Line Balancing Problem

DQN : Deep Q-Learning

QN : Q-Network

MSE : Mean square error

TD : Différence temporelle

PDM : processus de décision de Markov

INTRODUCTION GENERALE

L'industrie a vécu quatre révolutions et se prépare à la cinquième. La première révolution industrielle (Industrie 1.0) a été un changement majeur : les articles étaient produits par des machines. Le deuxième changement dans l'industrie manufacturière est appelé Industrie 2.0, qui a permis un transfert plus rapide des personnes et des idées novatrices. Cette révolution est une période de croissance économique, augmentant la productivité des entreprises. Ensuite, l'industrie 3.0 est appelée la révolution numérique en raison de l'automatisation des commandes à mémoire programmable et des ordinateurs. Actuellement, l'industrie 4.0 est une union entre les actifs physiques et les technologies avancées telles que l'intelligence artificielle, les robots, l'impression 3D, le cloud computing, etc. Les organisations qui ont adopté la technologie 4.0 sont flexibles et prêtes à prendre des décisions fondées sur des données. L'industrie 5.0 est la prochaine génération de technologies conçues pour des robots collaboratifs, efficaces et intelligents (Adel, 2022).

Dans l'industrie 4.0 et bientôt 5.0, la maintenance, la re-fabrication et le recyclage sont cruciaux du point de vue de la rentabilité et de la durabilité environnementale. L'émergence de l'intelligence dans les systèmes mécaniques au sein des industries nouvelles a facilité l'accès aux données en temps réel et assuré l'inter connectivité du système grâce à des pièces intelligentes intégrées à l'Assemblage 4.0 (A4.0). Cependant, cela à augmenter la complexité du désassemblage pour la maintenance ou la fin de vie (FV) de ces produits. Cela peut engendrer la nécessité d'intégrer les contraintes de démontage rapide et d'accessibilité des pièces dès les phases amont de la conception (spécification des exigences et des fonctions techniques).

Le problème du désassemblage est considéré parmi les problèmes NP-difficiles. En effet, le nombre de plans de désassemblage, qu'ils soient réalisables ou non, augmente de manière significative avec le nombre de pièces. Par conséquent, la génération d'un plan de désassemblage devient une tâche dure. Face à cette réalité, notre objectif est d'établir une méthodologie permettant de générer automatiquement des plans de désassemblage pour les produits mécaniques, qu'il s'agisse du désassemblage pour la maintenance préventive ou pour la fin de vie du produit. Il est également essentiel de démontrer la faisabilité des séquences de désassemblage ainsi générées. L'objectif de ce travail de thèse est de développer une méthodologie de génération des plans de désassemblage pour la maintenance, la fin de vie ou même pour la reconception des solutions pouvant avoir des défaillances de conception. L'intelligence artificielle est plus particulièrement l'apprentissage par renforcement sera utilisé pour l'optimisation des séquences de désassemblage générées.

Ce mémoire de thèse s'articule autour de quatre chapitres :

Le premier chapitre de cette thèse aborde une revue de la littérature visant à mettre en évidence l'importance de la planification des séquences de désassemblage dès la phase de conception et après utilisation du produit. Cette démarche nous permettra de récapituler les avantages et les inconvénients

de chaque approche, afin de définir clairement nos objectifs de recherche.

Le deuxième chapitre a pour objectif de mettre en œuvre la méthode d'apprentissage renforcé dans la génération des séquences de désassemblage. Dans un premier temps, les hypothèses de l'approche développée sont présentées ensuite une présentation des différentes étapes de l'algorithme développé sera dévoilée. Un exemple illustratif sera traité tout au long de cette partie afin de mieux expliquer les différentes étapes de la démarche proposée.

Le troisième chapitre aborde l'implémentation informatique de l'approche proposée pour la génération et l'optimisation des séquences de désassemblage. Plusieurs critères ont été considérés dans cette approche et la maintenance a été considérée comme le principal critère de cette étude. L'objectif essentiel de l'étude étant de générer une séquence de désassemblage faisable en se basant sur l'exploration des matrices d'interférence du modèle CAO. A la fin de ce chapitre, un exemple de validation issue de la littérature a permis de montrer les avantages de l'approche proposée, ainsi qu'une comparaison avec deux approches tirées de la littérature.

Le quatrième chapitre présente une approche de conception pour l'optimisation du processus de montage sélectif et de désassemblage (DFSA/D). Cette approche se concentre sur l'étude et la modification de l'architecture de conception d'un produit existant afin de faciliter l'accès aux pièces d'usure et de standardiser les fixations pour réduire le temps et les coûts de maintenance. La validation de l'approche proposée, est présentée à travers la planification d'une séquence de d'assemblage et de désassemblage sélectif d'un produit mécatronique.

Ce mémoire sera clôturé par une conclusion générale et des perspectives. Dans cette conclusion, nous récapitulons les principaux avantages et limitations de l'approche proposée. Ces limitations serviront de base pour nos perspectives de recherche dans de futurs travaux.

CHAPITRE 1 : ETAT DE L'ART ET OBJECTIFS DE RECHERCHE

1 INTRODUCTION : CONTEXTE

Afin de rester concurrentielles sur le marché, les industries manufacturières doivent optimiser trois paramètres principaux : le coût, la qualité et les délais. En prenant en compte les différentes phases du cycle de vie des produits, la planification des séquences de désassemblage des produits apparaît comme une activité critique impactant fortement ces paramètres (Kamble et al., 2020). En effet, dans la phase de maintenance, l'étude de la planification des séquences de désassemblage partiel (sélectif) peut contribuer à réduire les coûts de maintenance, à améliorer la qualité des produits. Par ailleurs, la fin de vie (End of Life) (EoL) des produits est également une étape dans laquelle la planification des séquences de désassemblage (totale) a maintenant une importance critique, notamment depuis la mise en place de la loi REP1 (Responsabilité Elargie des Producteurs) qui oblige les fabricants à intégrer l'écoconception de leurs produits, la prévention de leurs déchets, l'allongement de leur durée d'usage (en agissant sur le réemploi, la réutilisation, la réparation), et la gestion de leur fin de vie. Cette évolution règlementaire a une forte incidence que l'industrie manufacturière doit prendre en compte, tant dans les coûts que dans les pratiques industrielles de la conception à la fin de vie des produits (Cai and Choi, 2019).

Dans ce chapitre, notre objectif est de mettre en évidence l'importance de la planification des séquences de désassemblage tout au long du cycle de vie du produit. Nous commencerons par examiner les travaux clés dans le domaine du désassemblage des produits mécaniques, en nous appuyant sur une revue de la littérature. Cette démarche nous permettra d'analyser le besoin dans ce domaine et de récapituler les avantages et les inconvénients des approches existantes, pour définir les enjeux d'optimisation de la Planification des Séquences de Désassemblage (PSD) et identifier les pistes scientifiques apportées par les techniques d'intelligence artificielle pouvant y répondre, afin de définir les objectifs de recherche spécifiques de cette thèse.

2 ANALYSE DU BESOIN (MBSE)

L'Ingénierie des Systèmes Basée sur les Modèles (Model-Based Systems Engineering, MBSE) est une approche méthodologique qui utilise des modèles pour développer et gérer des systèmes complexes. Le MBSE présente plusieurs avantages pour l'analyse du besoin. Il permet une représentation claire et structurée des besoins, facilite la communication entre les parties prenantes. Il offre de nombreuses vues, tout en assurant leur cohérence, grâce à la traçabilité tout au long du processus de développement et ainsi facilite la gestion des changements (Henderson and Salado,

[1] https://filieres-rep.ademe.fr/

2021). Cette méthode est utilisée pour démontrer et clarifier les enjeux de la planification du désassemblage en soulignant les verrous et contraintes associés. Le langage de modélisation choisi pour faire cette analyse est SysML (System Modeling Language), langage de modélisation spécifique au domaine de l'ingénierie système. Il permet la spécification, l'analyse, la conception, la vérification et la validation de nombreux systèmes et systèmes-de-systèmes.

2.1 DEFINITION DU SYSTEME D'INTERET

La première vue à définir est le périmètre d'analyse qu'on souhaite adresser. Ici, nous avons plusieurs systèmes en interaction : le produit à désassembler, le processus de désassemblage, la planification à optimiser. Nous avons représenté ces systèmes sur des diagrammes de définition de bloc (bdd).

Dans la Figure 1, le produit assemblé peut être de nature mécanique ou mécatronique. Un assemblage mécatronique peut générer des contraintes supplémentaires sur le désassemblage (passage de câbles, sauvegarde préalable de la configuration …) mais peut présenter aussi des avantages (capteurs et actionneurs permettant de faciliter le désassemblage). Un assemblage (produit) est composé de pièces et d'éléments de fixation déterminant si l'assemblage est réversible et donc si le désassemblage est ou pas destructif. Enfin, il est en interaction avec 3 types d'acteurs : (i) le concepteur qui est en charge de concevoir et reconcevoir (après le retour d'expérience des utilisateurs/experts et de la maintenance) le système et/ou son assemblage pour faciliter son désassemblage, (ii) l'agent de maintenance identifie les pièces d'usure qui doivent être démontées facilement (donc avec un accès facile et direct), (iii) l'expert permet de faire remonter son retour d'expérience et les bonnes pratiques. Enfin, le produit assemblé est en interaction avec le/les moyens de désassemblages, qui comprennent les outils, les moyens de déplacement de la pièce (changement de direction, transport), et peuvent être accomplis soit par un opérateur humain soit par un système automatisé.

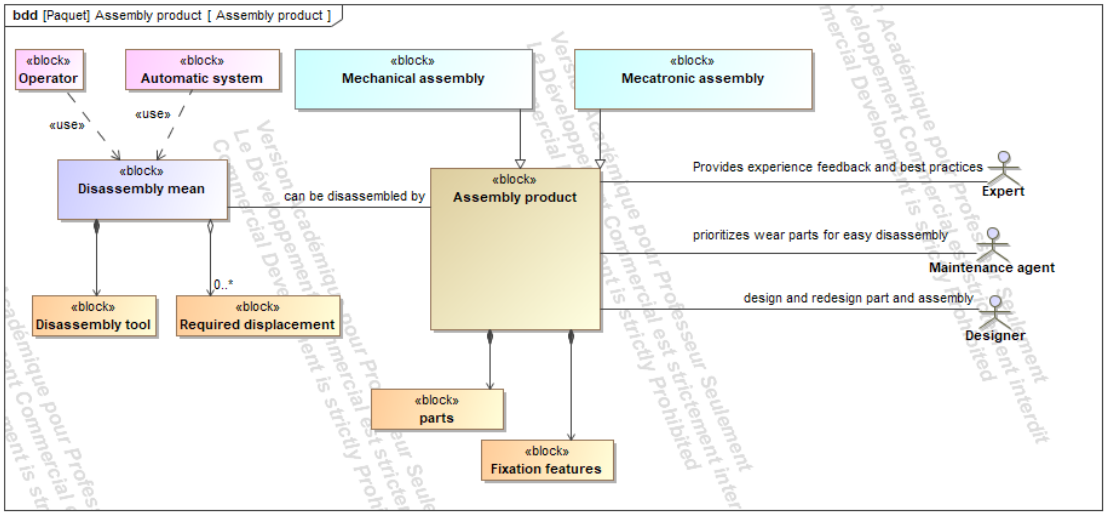


Figure 1 : Représentation du produit assemblé

La Figure 2 présente le processus de désassemblage qui a un niveau et un type de désassemblage. Le niveau de désassemblage adresse la notion de désassemblage partiel/sélectif ou complet. Concernant son type, il peut être de nature séquentielle ou parallèle (plusieurs pièces sont désassemblées en même temps) en mode ou non coopératif.

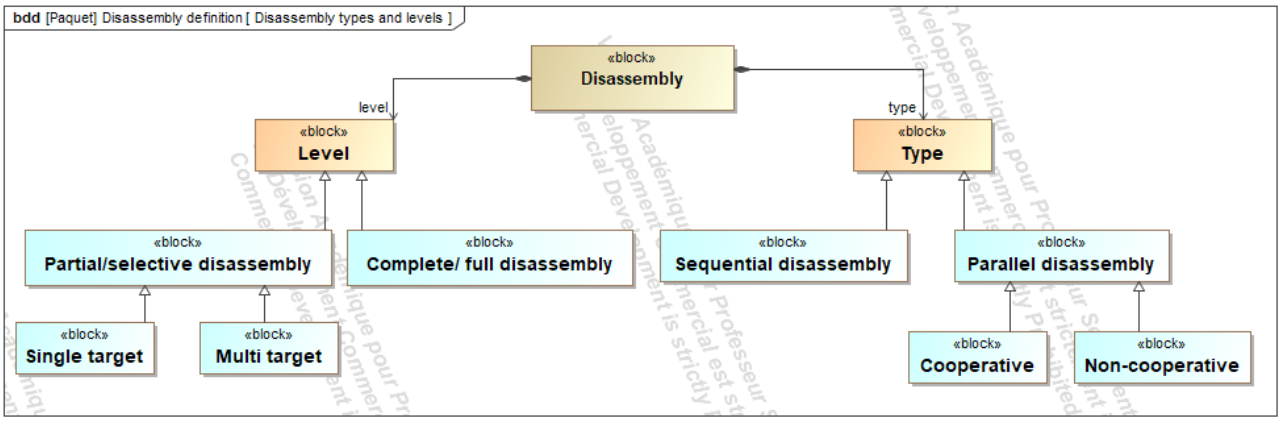


Figure 2 : Représentation du processus de désassemblage.

La Figure 3 décrit l'influence du produit assemblé et du processus de désassemblage sur l'optimisation de la planification des séquences de désassemblage. Cette optimisation du PSD peut avoir un ou plusieurs objectifs, qui se déclinent chacun en un ou plusieurs critères. Elle se caractérise par son efficience à remplir l'objectif avec le minimum de ressources.

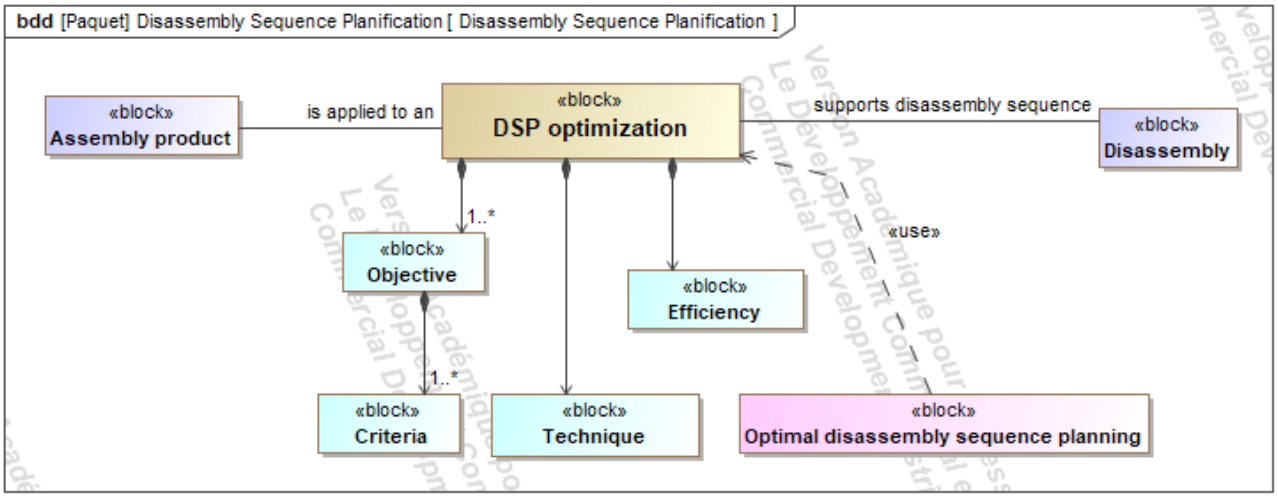


Figure 3 : Représentation de l'optimisation des séquences de désassemblages

2.2 CYCLE DE VIE DU PRODUIT ASSEMBLE

Le cycle de vie d'un produit assemblé (Figure 4) se compose de plusieurs phases, débutant par la conception, qui définit les spécifications et les exigences du produit, puis le développement, au cours duquel l'architecture de l'assemblage est définie, que nous avons regroupées dans une phase de

« Design », qui du point de vue de la problématique de désassemblage, se finit lorsque la conception a été validée et que le modèle de l'assemblage 3D est terminé. Les composants sont alors fabriqués, assemblés et soumis à des tests de qualité et la production en série commence, et le produit est utilisé par les consommateurs tout en bénéficiant d'une maintenance régulière, qui se traduit par un désassemblage partiel. À la fin de sa vie, le produit doit alors être recyclé pour permettre la réutilisation de ses pièces, en s'appuyant sur un désassemblage total. Si lors de la maintenance ou de la phase de fin de vie, le processus de désassemblage n'est pas satisfaisant, le produit passe en reconception pour optimiser la conception de l'assemblage en vue d'améliorer les processus de désassemblage partiel et total et notamment sa planification.

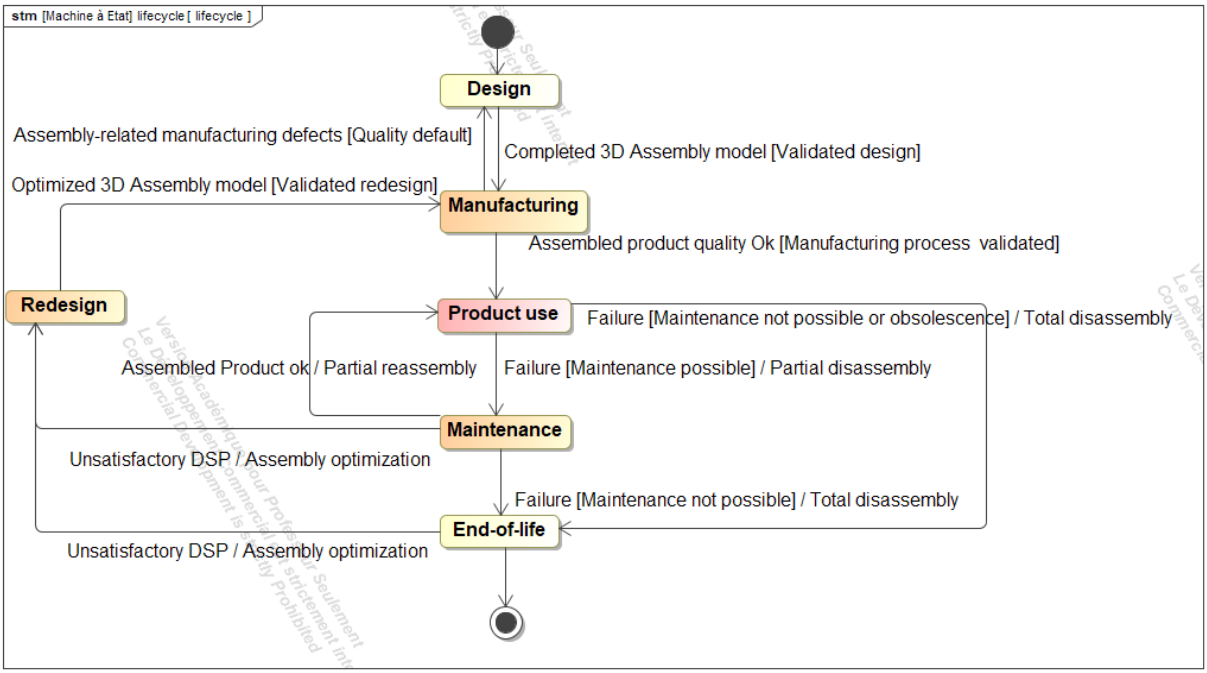


Figure 4: Cycle de vie d'un produit assemblé, du point de vue de notre problématique de désassemblage

Les problématiques de la génération des séquences de désassemblage partiel et total se retrouvent donc intégrées aux phases de conception, maintenance, fin de vie et reconception.

2.3 PROCESSUS DE PSD

En se focalisant sur le processus d'optimisation des séquences de désassemblage, la plupart des méthodes présentes dans la littérature suivent le même processus générique. Le processus commence par extraire les informations du produit lui-même, que ce soit en examinant le produit physique ou en analysant son modèle CAO 3D. Cette extraction peut être effectuée manuellement ou automatiquement. Ensuite, il s'agit de choisir et d'implémenter une représentation du produit qui mette en évidence les relations entre les pièces et leur mode de fixation. Ensuite, il est nécessaire de générer des séquences de désassemblage réalisables, avant de déterminer la séquence de

désassemblage optimale au regard d'un ou plusieurs paramètres à définir, et choisir la méthode d'optimisation appropriée.

2.4 EXIGENCES DERIVEES

L'analyse des diagrammes précédents a permis de générer et structurer des exigences dérivées pour présenter une vue d'ensemble des aspects et contraintes inhérents à la problématique d'optimisation de séquence de désassemblage d'un produit. Ces exigences dérivées sont présentées dans la Figure 5.

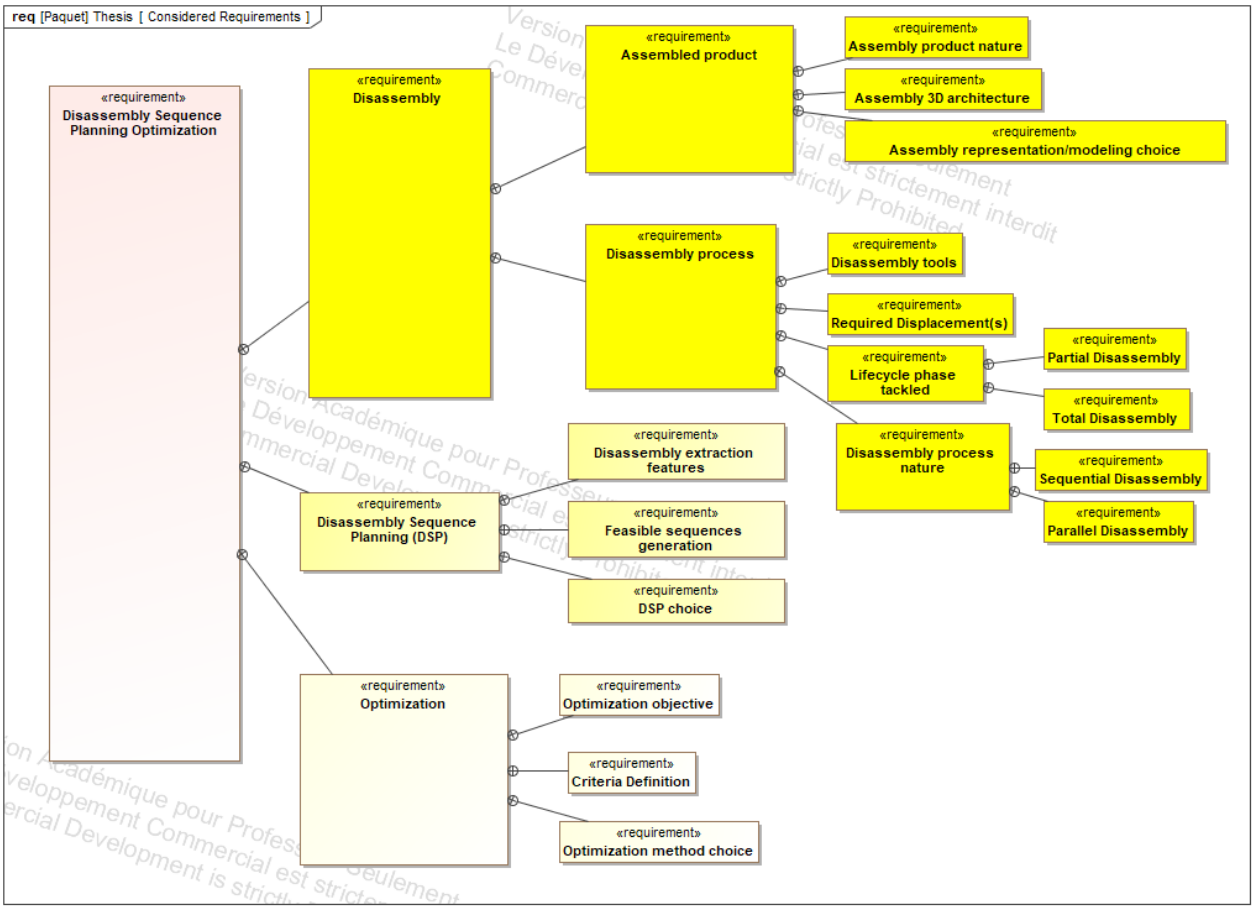


Figure 5 : Exigences dérivées décrivant le besoin et son contexte.

Ces exigences nous ont permis de cibler et structurer notre état de l'art puis d'identifier les verrous qui seront adressés dans cette thèse.

3 LA PLANIFICATION DES SEQUENCES DE DESASSEMBLAGE

Initialement les premiers travaux de recherche portant sur la planification des SD étaient uniquement théoriques et totalement déconnectés des plateformes de CAO. La détermination des Séquences de Désassemblage (**SD**) a attiré l'attention de nombreux chercheurs au cours des dernières

années. Ces approches consistent à établir le plus automatiquement possible, les SD d'un mécanisme à partir d'un modèle de conception assistée par ordinateur **CAO** ou d'un produit déjà existant. L'avantage de ces approches est non seulement de gagner du temps en générant les SD mais aussi de fournir un outil de validation de la conception (en évitant un désassemblage impossible par exemple).

Les verrous scientifiques liés à la génération optimisée des SD adressent les problématiques suivantes : (i) le choix de la méthode de modélisation du produit assemblé selon le niveau de complexité de son architecture 3D, (ii) la caractérisation des paramètres influents garantissant l'adéquation d'une séquence et finalement (iii) le choix de la méthode d'optimisation et la génération d'une séquence optimale.

3.1 LES METHODES DE MODELISATION DES PRODUITS ASSEMBLES

Dans la littérature, pour l'arrangement des SD, on distingue deux méthodes de modélisation des produits mécaniques : la modélisation par la méthode des graphes et la modélisation par des matrices. Cette section décrit la modélisation basée sur les graphes et la modélisation basée sur les matrices (Huang and Xu, 2017). La modélisation des assemblages est essentielle pour la définition du séquencement et des opérations de désassemblage des produits. En effet, certains facteurs influencent la faisabilité d'une SD, tels que les liaisons cinématiques entre les composants, les contraintes industrielles liées au retrait d'un composant, leur forme géométrique, un aspect de désassemblage présentant un risque (détérioration, sécurité…), les outils nécessaires aux opérations de désassemblage, l'accessibilité de l'espace pour les outils et la complexité (géométrique, accès, nombre de composants, etc.) du produit à assembler et des fixations à retirer pour désassembler les composants cibles. La faisabilité d'une séquence de désassemblage dépend donc de la structure et architecture géométrique du produit.

3.1.1 Modélisation par les méthodes de graphes

3.1.1.1 Modélisation par les graphes ET/OU

La modélisation par graphe s'appuie sur des graphes orientés et non orientés, composés d'un ensemble de nœuds et d'arêtes. Les premiers travaux d'intégration de cette modélisation dans les approches de PSD commencent en 1988, lorsque Homem de Mello et Sanderson proposent une approche basée sur les graphes ET/OU qui permet de représenter les relations entre les composants et les sous-assemblages d'un produit par plusieurs arcs (Homem De Mello and Sanderson, 1988). En 1992, Dini et Santochi ont utilisé ce graphe ET/OU pour représenter les plans d'assemblage (**PA**) pour les mécanismes complexes, en transformant le PA en un graphe (Dini and Santochi, 1992). Dans ce contexte, un nœud représente un assemblage ou un composant (qui peut être un sous-ensemble) et les arêtes reliant les nœuds représentent les opérations entre les différents composants, que cette modélisation soit utilisée dans un processus d'assemblage ou de désassemblage, la conversion de ces 2 problèmes peut facilement se faire sous réserve que les opérations de d'assemblage soient réversibles. Ainsi, à partir des contraintes de précédence définies entre les nœuds, la planification des

séquences possibles de désassemblage d'un produit peut être vue comme une recherche de chemin exprimées par un graphe orienté. La séquence de désassemblage optimale peut alors être identifiée manuellement ou en utilisant des algorithmes sur la base de ces graphes.

Le principe du graphe ET/OU utilisé (qui est en fait un hypergraphe), est basé sur la modélisation des assemblages où les nœuds terminaux sont les composants unitaires et les hyper-arcs (k-connecteur avec ($k \epsilon N^*$) correspondent aux opérations de désassemblage. Chaque hyper-arc qui quitte un nœud correspond à une opération de désassemblage applicable à l'assemblage de ce nœud, et les nœuds suivants vers lesquels pointe l'hyper-arc correspondent aux sous-ensembles produits (relation ET) par l'opération de désassemblage. Chaque nouveau point de départ d'un hyper-arc représente une autre alternative de désassemblage (relation OU) (Tang, 2002).

Depuis, de nombreux chercheurs ont exploité le modèle de graphe ET/OU pour la génération des SD. Par exemple, Zhang et Kuo ont développé une méthode pour automatiser la transformation de la représentation 3D d'un produit en un graphe, en s'appuyant sur l'identification des éléments de fixation, pour la planification des séquences de désassemblage en fin de vie. La représentation des relations entre les pièces mécaniques permet de générer un graphe approprié pour définir la séquence de désassemblage total (en fin de vie) optimale vis-à-vis du coût de recyclage (Zhang and Kuo, 1997). Avec le développement informatique des plateformes de conception, les chercheurs ont commencé à travailler directement sur les PSD en utilisant les plateformes 3D pour le démantèlement des produits (De-Manufacturing (**DM**)) défini comme un processus de désassemblage de certaines pièces ou de certains composants d'un produit. Ces pièces ou composants peuvent être sélectionnés pour être recyclés, réutilisés, entretenus ou éliminés (Chung and Peng, 2005). Les SD ont été générées automatiquement pour la première fois, par le biais du graphe de liaison hiérarchique attribuée (Hierarchical Attributed Liaison Graph, (HALG)), en démontant d'abord tous les sous-ensembles, puis les pièces (Dong et al., 2006).

Le graphe ET/OU est l'un des graphes les plus utilisés pour modéliser les relations entre les composants. On peut par exemple citer le modèle de graphe hybride de désassemblage qui met en œuvre la transformation d'un produit réel en un modèle mathématique qui contient des informations sur les relations de contrainte. En utilisant le modèle Hybrid Disassembly Graph Model **(HDGM)**, Zhang et Zhang ont proposé une méthode d'optimisation de PSD par l'optimisation du temps de désassemblage. Les auteurs ont réussi à incorporer des contraintes mécaniques dans les graphes de désassemblage pour minimiser le temps de désassemblage, mais cette méthode ne concerne qu'une seule étude de cas et n'a pas été généralisée pour être appliquée à n'importe quel assemblage mécanique (Zhang and Zhang, 2010). De même, Wang et al, ont utilisé la méthode des graphes pour représenter les contraintes entre les composants, et notamment celles nécessitant un désassemblage destructif ou non, dans le cas d'un désassemblage de fin de vie d'un bras mécanique (les fixations doivent être retirées avant de désassembler les composants associés). Chaque composant est représenté par un cercle avec un numéro de composant à l'intérieur. Si deux composants sont en contact, ils sont reliés par une ligne continue. Une ligne continue avec une flèche indique les

contraintes liées aux fixations qui peuvent être supprimées par des opérations de désassemblage non destructives. Une ligne en tirets avec flèche indique les contraintes liées aux fixations qui ne peuvent être enlevées que de manière destructive (Wang et al., 2017). De leur côté, Issaoui et al. ont proposé une méthode de génération automatisée de séquences de désassemblage d'un produit mécanique à partir d'un modèle CAO, basée sur un arbre de connexion approprié d'un composant ciblé. Cet arbre vise à réduire l'espace de solutions par des règles d'élimination. La génération de séquences est fondée sur la lecture des branches de l'arbre de connexion et par élimination sur des tests de faisabilité. La faisabilité est vérifiée par la mise à jour de la mobilité du désassemblage de chaque élément de la séquence (Issaoui et al., 2015).

3.1.1.2 Modélisation des réseaux de Pétri

Les réseaux de Pétri (Petri Net (**PN**) ont été utilisés pour modéliser la relation structurelle d'un produit ou d'un système assemblé. En effet, la représentation PN est un outil populaire dans la modélisation systémique (Zhou and DiCesare, 2012; Zhou and Venkatesh, 1999). Elle peut traiter les relations de précédence entre les actions. Ainsi Guo et al. ont proposé un PN de Désassemblage (**PDN**) qui est composé de jetons, de places, de transitions et d'arcs avec des règles d'exécution. Un jeton représente la disponibilité d'un produit, sous-ensemble ou composant. Les places désignent les sous-ensembles ou composants, et les conditions sont associées aux opérations de désassemblage. Les transitions correspondent aux actions de désassemblage, et les arcs sont similaires aux hyper-arcs dans les graphes ET/OU pour les itinéraires de désassemblage.

Définition : Un PDN est défini comme un septuplé PDN = (P, T, I, O, M, c, τ) où :

- P est un ensemble de places.
- T est un ensemble de transitions avec P\T = ∅ et P[T , ∅].
- I: P × T → N est une fonction d'entrée qui définit un ensemble d'arcs dirigés de P à T.
- O: O × T → N est une fonction de sortie qui définit un ensemble d'arcs dirigés de T à P.
- M: P → N est un marquage représentant le nombre de jetons dans chaque place. Le marquage initial est noté M0.
- c: T → R+ est un coût de désassemblage associé à une transition, où R+ est l'ensemble des nombres entiers non négatifs.
- τ: P → R+ est une valeur de recyclage/réutilisation associée à une place.

L'exemple d'un PDN d'un stylo à bille est présenté à la Figure 6.

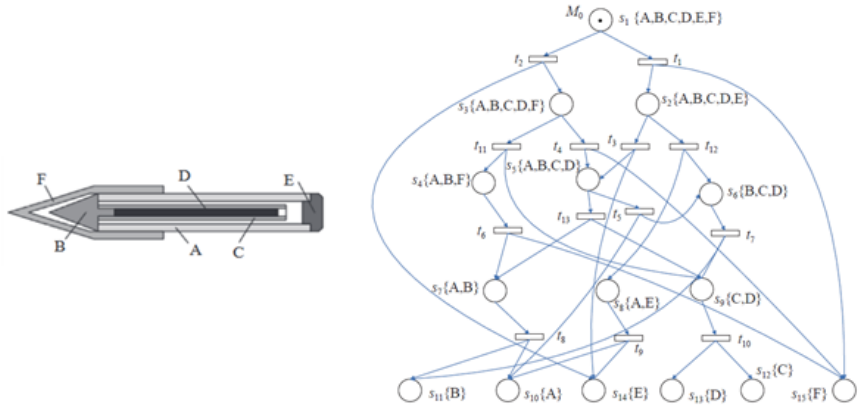


Figure 6: Un PDN d'un stylo à bille (Guo et al., 2021)

En utilisant cette méthode, une SD est déterminée en fonction du coût et des avantages environnementaux (recyclage, récupération des pièces) d'un processus de désassemblage. L'intérêt et l'efficacité de la méthode proposée ont été illustrés sur une étude de cas réel. Mais comme un modèle basé sur un graphe, le PDN devient très complexe lorsque le nombre de composants du produit est élevé, et il est difficile de le générer automatiquement à partir de plateformes numériques de conception. Ceci se traduit par une complexité de calcul élevée lors de la recherche de plans de désassemblage optimaux.

3.1.2 Modélisation matricielle

Dans les opérations de désassemblage, les composants sont principalement enlevés par des mouvements de translation, tandis que certains composants, tels que les vis et les boulons, sont retirés par des mouvements de rotation le long de leurs axes centraux (Pornsing and Watanasungsuit, 2014). Pourtant, lors de la planification du désassemblage, tous les mouvements sont modélisés comme des mouvements de translation. Par conséquent, dans un système en coordonnées cartésiennes, seuls trois Degrés De Liberté (**DDL**) sont généralement pris en compte parmi les six directions de désassemblage : ±X, ±Y, ±Z (Zhu et al., 2013). Pour une opération de désassemblage sélectif, l'opération de désassemblage s'arrête dès que le composant cible est désassemblé (Luo et al., 2016). Pour les produits ayant une structure complexe et un grand nombre de composants, la planification de la séquence de désassemblage pour un composant cible est difficile et la solution optimale est risque de ne pas être atteinte (Pomares et al., 2004). Pour permettre l'opération de désassemblage, les types de contraintes entre les composants sont déterminés et de préférence traitées par des opérations non destructives, excepté pour certaines contraintes comme le rivetage et le soudage.

Pour pallier aux inconvénients de la modélisation par graphes lorsque le nombre de composants du produit à étudier est élevé, les chercheurs ont développé de nouvelles méthodes de représentation telles que les représentations matricielles qui peuvent intégrer la structure des produits, les connexions des composants ainsi que les relations de contraintes dans des matrices (ElSayed et al., 2012). Comparées à la modélisation par graphe, les matrices sont plus pratiques et plus flexibles car

intégrables pour le traitement informatique (Zhang Gang et al., 2006). Par conséquent, la modélisation matricielle est couramment utilisée en PSD pour représenter les connexions géométriques et les contraintes mécanique des produits. Ainsi, par exemple la matrice de contraintes peut être utilisée pour représenter les données des contraintes entre les composants et notamment celles des opérations de désassemblage (Yu et al., 2014).

La modélisation matricielle des produits mécaniques consiste à présenter les relations pièce/pièce, sous-ensembles/sous-ensemble ou pièce/sous-ensemble en utilisant un ensemble de matrices selon les trois directions de l'assemblage (X, Y, Z), telles que : les matrices de contact, la matrice de connexion, les matrices d'adjacence, matrice d'interférence, etc., dont les principales sont présentées en ***Annexe 1.*** Ces matrices ont été intégrées dans un outil informatique pour générer automatiquement le diagramme de désassemblage. Ce diagramme présente dans chaque nœud la pièce à démonter en précisant la direction de montage et le reste des pièces qui restent assemblées. Le processus se répète de manière itérative jusqu'à qu'une seule pièce reste dans le groupe des pièces à démonter.

Concernant les travaux de recherche basés sur la modélisation matricielle, Santochi et al. ont proposé une représentation matricielle manuelle des liens en utilisant les codes binaires pour générer les Séquences de Désassemblage (SD)(Santochi M, Dini G 1991). Une approche de PSD pour le recyclage et la récupération des pièces défectueuses a été proposée par Gungor et Gupta, où les pièces démontées peuvent être remises à neuf, pour améliorer la récupération des machines dans la phase de fin de vie (Gungor and Gupta, 1998a). Dans les années 2000, les auteurs ont commencé à se concentrer sur l'intégration des contraintes mécaniques dans les représentations de PSD des produits. En utilisant pour la première fois l'algorithme génétique (GA) pour la planification des séquences de désassemblage et de maintenance (MSP), les auteurs ont obtenu des résultats encourageants pour la planification des séquences de désassemblage (Li et al., 2002) (Hao, 2008) (Tseng et al., 2010). Les matrices représentant les relations entre les pièces mécaniques dans les assemblages ont été le moteur de l'automatisation du PSD.

La section suivante présente une vue d'ensemble de ces différentes approches de PSD intégrant différents modes (destructif ou non destructif), niveaux (partiel ou complet) et types (séquentiel ou parallèle) de désassemblage et objectifs d'optimisation associés.

3.2 LA GENERATION DES SEQUENCES DE DESASSEMBLAGE

Les processus de désassemblage peuvent être *destructifs* et *non destructifs*, en fonction des objectifs du désassemblage, qui peuvent varier suivant les phases du cycle de vie d'un produit. Lors d'un désassemblage non destructif, une pièce est récupérée dans son ensemble pour être réparée, réutilisée, remise à neuf, etc. Dans le cas d'un désassemblage destructif, la pièce peut être endommagée lors de l'extraction, puisque souvent l'objectif porte plutôt sur la récupération des matériaux. En général, le désassemblage non destructif est préférable car le désassemblage destructif est irréversible. Le désassemblage destructif ne devrait être effectué que lorsque la séparation des composants nécessite la destruction de certains d'entre eux ou lorsque le coût du désassemblage non

destructif est supérieur à celui du désassemblage destructif (Özceylan et al., 2019).

En fonction de l'objectif du désassemblage et des exigences de l'utilisateur, le problème du PSD peut porter sur un désassemblage *manuel* ou un désassemblage *automatisé* (Liu et al., 2018). Certains processus de désassemblage complexes de petites pièces peuvent être impossibles à réaliser par un robot, tandis que d'autres opérations de désassemblage destructif peuvent être trop dangereuses pour un humain et doivent être automatisées. Les processus et environnements de désassemblage collaboratif homme-robot ont donc été étudiés pour adresser cet enjeu (Liu et al., 2019).

En fonction de la profondeur de désassemblage, la SD peut être classée en différents niveaux : désassemblage *complet* ou *sélectif/partiel*. Dans le cas du désassemblage complet, chaque pièce peut être retirée puisque le produit doit être entièrement désassemblé. Cette opération est généralement effectuée sur les produits en fin de vie (**EoL** en anglais pour End Of Life). Le désassemblage partiel est généralement utilisé pour récupérer des pièces spécifiques tout en laissant les autres intactes (Ghandi and Masehian, 2015). Il est fréquemment utilisé dans les processus d'échange de pièces et de maintenance, lorsque seules certaines pièces doivent être maintenues. La planification du désassemblage partiel peut être divisée en deux autres catégories : la planification du désassemblage à *objectif unique* et la planification du désassemblage à *objectifs multiples*. La planification du désassemblage à objectif unique recherche les séquences possibles pour restaurer une seule pièce cible, tandis que la planification du désassemblage à objectifs multiples impose des contraintes supplémentaires afin que plusieurs pièces cibles puissent être retirées dans la même séquence de désassemblage (Meng et al., 2016).

Enfin, le type de désassemblage peut être *séquentiel* ou *parallèle* (Smith and Hung, 2015). Le désassemblage séquentiel consiste à démonter les pièces une par une, et le désassemblage parallèle consiste à démonter plusieurs pièces en même temps. Cela peut se faire en désassemblant simultanément de nombreuses pièces sous forme de sous-ensembles ou en permettant à deux opérateurs humains ou plus de coopérer et de désassembler plusieurs pièces en même temps. Lorsque plusieurs opérateurs effectuent différentes étapes de désassemblage en parallèle sur le même produit, on parle de *désassemblage coopératif*. Les systèmes de désassemblage coopératif peuvent inclure à la fois la coopération homme-robot et la collaboration humaine. Ces différentes catégories de désassemblage sont synthétisées dans la Figure 7.

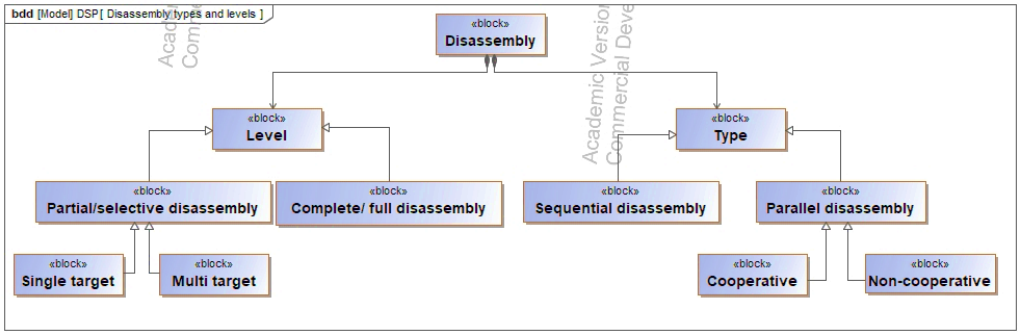


Figure 7: Les types et les niveaux de désassemblage

Après les phases de fabrication et d'assemblage final, la SD pour la maintenance et la SD pour la fin de vie sont les phases les plus coûteuses du cycle de vie du produit. Les principaux objectifs d'optimisation de la SD portent alors généralement sur la réduction du coût et du temps de désassemblage, qui peut être optimisé en prenant en compte des critères supplémentaires: la minimisation du nombre d'outils et des changements de direction du désassemblage, etc. (Gungor and Gupta, 1998a; Issaoui et al., 2015; Ghandi and Masehian, 2015; Guo et al., 2021).

L'optimisation de la SD a fait l'objet de nombreuses études, dont nous allons maintenant présenter le processus général. En effet, indépendamment du type de désassemblage, du mode et des paramètres d'optimisation, chaque optimisation de séquence de désassemblage nécessite les étapes présentées dans la Figure 8.

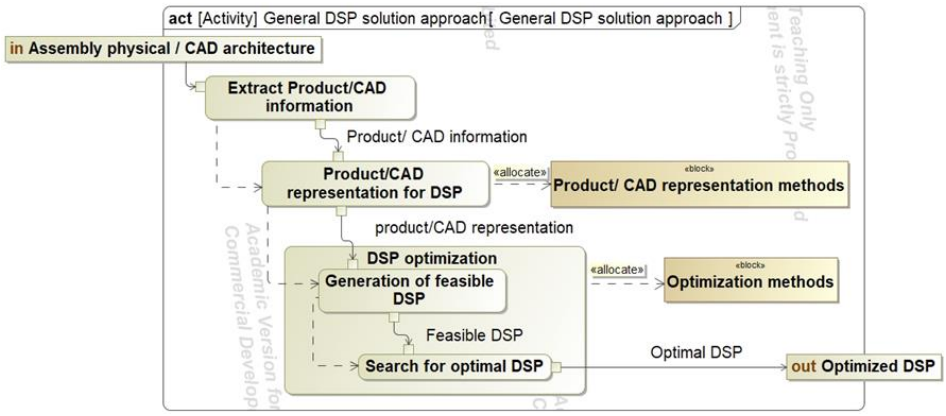


Figure 8. La démarche d'optimisation des séquences de désassemblage

La première étape consiste à extraire les informations du produit lui-même, soit à partir du produit physique, soit par l'analyse de son architecture CAO ; cette extraction peut être manuelle ou automatisée. La deuxième étape consiste à choisir une représentation de l'assemblage qui montre clairement les relations entre les pièces et la manière dont elles sont attachées et à le modéliser avec cette représentation. Troisièmement, les séquences de désassemblage réalisables doivent être générées. Enfin, la séquence de désassemblage optimale doit être identifiée en fonction d'un ou de plusieurs des paramètres d'optimisation mentionnés précédemment, en choisissant une méthode d'optimisation.

4 OPTIMISATION DE LA PLANIFICATION DES SEQUENCES D'ASSEMBLAGE

4.1 OBJECTIFS ET CRITERES D'OPTIMISATION DE PSD

L'optimisation des SD est un objectif industriel critique compte tenu de son impact important en termes de coûts et de temps de maintenance et de démantèlement. Ces deux objectifs peuvent être

atteints en optimisant plusieurs critères pour générer la séquence de désassemblage.

Le premier objectif d'optimisation des SD est de minimiser le temps de réalisation de la séquence. Ce critère dépond de plusieurs paramètres liés à la réalisation pratique de la séquence, parmi ces paramètres on cite : (i) le nombre de changement d'outils (tournevis, presse etc.) durant l'opération du désassemblage, ainsi que le nombre de changements de direction de désassemblage des pièces, (ii) la mise en place d'équipements spécifiques/instrumentations industriels, un bon agencement de l'armoire du magasin d'outils et de la table de désassemblage peuvent minimiser les temps d'aller-retour à chaque changement d'outil, (iii) la difficulté de manipulation du système pendant le désassemblage, (iv) la difficulté d'accès au pièces de maintenance etc. Tous ces paramètres impactent directement la durée de désassemblage physique intégrant les délais de préparation des systèmes lors de désassemblage pour la maintenance, et par conséquence son coût global : main d'œuvre nécessaire, outillages, déplacement des pièces…

Le deuxième objectif d'optimisation porte sur la minimisation du temps de génération des SD. Ce paramètre est directement lié à l'aspect algorithmique de la méthode d'optimisation choisie et au temps d'exécution informatique (Zhou et al., 2019).

4.2 LES TECHNIQUES TRADITIONNELLES

La génération des SD est une étape cruciale du processus de désassemblage qui implique l'identification d'un plan de désassemblage optimal pour retirer des pièces spécifiques d'un mécanisme. Ainsi Gungor et Gupta ont proposé, sur la base de la matrice des priorités, un plan de désassemblage des produits pour le recyclage et la récupération des pièces défectueuses, dans lequel les pièces remises à neuf peuvent être démontées (Gungor and Gupta, 1998b). Les travaux de recherche se sont ensuite concentrés sur l'intégration des contraintes mécaniques dans les représentations de planification des séquences de désassemblage des produits. Parmi les approches basées sur les données CAO qui proposent une optimisation basée sur la minimisation du nombre de changements d'outil pendant l'exécution de la séquence d'assemblage, Bedeoui et al. ont réussi à aller plus loin concernant la visualisation 3D, en prenant en compte les outils d'assemblage dans la procédure d'assemblage. Ils ont proposé une approche dans laquelle ils introduisent le concept d'espace de travail des outils d'assemblage pour simuler et sélectionner la séquence d'assemblage optimale.

Pour le désassemblage coopératif, Belhadj et al. ont développé une approche basée sur la matrice d'interférence qui vise à intégrer la collaboration homme-robot (HRC). La méthode développée ajoute deux aspects : le parallélisme des tâches pendant le désassemblage et l'intégration des robots dans l'industrie. Cette approche se concentre sur le désassemblage EoL de machines complexes comportant un grand nombre de pièces (Belhadj et al., 2022).

Pour spécifier la SD optimale, Aicha et al. ont proposé une nouvelle approche qui combine deux métriques principales, l'indice de qualité (Qi) et l'indice de temps (Ti), comme critères de sélection

de la SD optimale et réalisable. Qi est calculé sur la base de la méthode d'analyse des modes de défaillance, de leurs effets et de leur criticité (AMDEC) du produit, et Ti représente l'indice de temps de traitement réel, en référence aux contraintes de fabrication (lieu de travail, disposition, flux de travail, outils, machines...) (Aicha et al., 2022, 2021). Le choix du plan de désassemblage optimal se base sur ces 2 critères Ti et Qi : le PD ayant le score Qi le plus élevé et le score Ti le plus bas est le plus optimal en se basant sur les outils de désassemblage, le changement de direction et le temps réel de traitement.

La plupart des approches citées précédemment sont réalisées avec des outils de programmation classiques qui nécessitent un temps d'exécution important qui peut varier d'une journée à une semaine de simulation. Avec le développement des outils informatiques et pour l'intégration dans les usines de plus en plus intelligentes et connectées, l'optimisation des PSD s'oriente vers les techniques d'intelligence artificielle.

4.3 LES ALGORITHMES BASES SUR L'IA

Une autre voie pour générer les PA/PD est de de considérer, dans une première étape, toutes les séquences possibles entre les pièces, y compris celles qui ne sont pas optimales ou réalisables. Puis, dans une deuxième étape, les séquences non optimales ou non réalisables sont éliminées. Pour cela, les chercheurs ont utilisé différentes méthodes basées sur les techniques d'intelligence artificielle, telles que les algorithmes génétiques et les algorithmes de colonies de fourmis et récemment basée sur les réseaux de neurones.

4.3.1 Les algorithmes génétiques

Les algorithmes génétiques exploitent des méthodes de recherche inspirées du processus évolutif naturel, telles que le croisement et la mutation. Pour générer les séquences de désassemblage, cette approche débute en créant une population initiale d'individus en utilisant des chromosomes, qui représentent dans ce cas une séquence de désassemblage aléatoire. Ensuite, la population est évaluée à l'aide d'une fonction objective qui prend en compte plusieurs critères d'optimisation. Les séquences obtenues sont ensuite modifiées en utilisant deux types d'opération génétique : le croisement et la mutation. Par la suite, les solutions obtenues sont réévaluées jusqu'à ce que les solutions optimales soient trouvées. Dans la plupart des études menées par (Chen and Liu, 2001), (Marian et al., 2006) et (Kheder et al., 2015a), les gènes représentent les composants de l'assemblage mécanique. Ainsi, chaque chromosome (séquence de désassemblage) peut être généré automatiquement ou défini manuellement par le planificateur. Il décrit une séquence complète de désassemblage et peut contenir plusieurs gènes correspondant aux pièces du produit.

4.3.2 Algorithmes des colonies de fourmis

L'Algorithme des Colonies de Fourmis (ACO) est une méthode innovante qui s'inspire du comportement naturel des fourmis pour optimiser des problèmes combinatoires, notamment le

problème de PSD. L'ACO est un algorithme itératif où tous les individus partagent une connaissance commune et collaborent de manière coopérative pour guider d'autres agents dans leurs choix. Cette collaboration se traduit par une communication indirecte entre les fourmis, réalisée par le dépôt d'une substance chimique appelée phéromones. Les fourmis ont pour objectif de rechercher un chemin optimal entre leur nid et la source de nourriture. Ainsi, en appliquant cette méthode au problème de génération des séquences de désassemblage, l'ACO vise à trouver un chemin optimal dans un graphe afin d'obtenir une séquence optimale se basant sur le comportement des fourmis. L'ACO utilise des probabilités basées sur les niveaux de phéromones déposés sur les arêtes du graphe. Les fourmis se déplacent dans le graphe en privilégiant les arêtes avec une probabilité plus élevée en fonction des niveaux de phéromones. Les chemins contenant une plus grande quantité de phéromones deviennent ainsi plus attractifs pour les fourmis suivantes. Au fil des itérations, l'ACO renforce les chemins les plus courts en déposant davantage de phéromones, ce qui conduit à une convergence vers une solution optimale.

En utilisant l'ACO, Failli et Dini, ont utilisé une méthode basée sur un graphe complexe appelé Réseaux de Nœuds de Désassemblage (RND) afin de représenter tous les schémas de désassemblage possibles dans le contexte du recyclage. L'utilisation de l'ACO (Ant Colony Optimization) pour le PSD présente l'avantage de réduire l'espace de recherche à explorer (Failli and Dini, 2001). D'autres recherches sur l'optimisation des PSSD en utilisant l'optimisation avec l'ACO dans le contexte du recyclage et de la réutilisation des pièces ont été développées (Fan et al., 2013). Aussi l'ACO est souvent utilisé pour corriger les défauts de l'AG (Wang et al., 2005). Malik, a développé l'ACO pour imiter le comportement des fourmis. Pour la PSD, si un composant dans une séquence de désassemblage ne peut pas être désassemblé, cette séquence de désassemblage serait marquée comme une séquence non réalisable et il n'est alors plus nécessaire de prendre en compte les composants suivants dans cette séquence de désassemblage (Malik, 2010). En utilisant l'ACO, Luo et al, ont proposé une nouvelle méthode pour intégrer la représentation multicouche et la recherche d'une planification quasi-optimale de la séquence de désassemblage pour le désassemblage sélectif. Cette approche a démontré sa faisabilité en étant appliquée à des produits industriels pour générer le PSD multicouche (Luo et al., 2016). La prise en compte de la maintenance d'un produit est un des aspects également abordés par Wang et al. qui ont appliqué l'ACO à l'optimisation des SD sélectives en spécifiant une liste cible des composants à maintenir (Wang et al., 2003).

4.3.3 Optimisation par essaim particulaire (Particule Swarm Optimisation)

L'optimisation par essaim particulaire (Particle Swarm Optimization, (PSO)) est un algorithme d'optimisation inspiré du comportement social des essaims d'oiseaux ou de poissons. Il peut être utilisé pour résoudre des problèmes d'optimisation dans divers domaines, y compris la génération de séquences de désassemblage. L'essaim de particules est une population d'agents ou de particules, où chaque particule représente une solution du problème avec une position et une vitesse. Chaque particule possède également une mémoire pour se souvenir de sa meilleure performance en termes de position et de valeur, appelée « information locale ». L'algorithme PSO est similaire à un algorithme

génétique (AG) dans le sens où il débute avec une population de solutions initiales aléatoires. Cependant, contrairement à l'AG, le PSO n'utilise pas d'opérateurs d'évolution tels que le croisement et la mutation. Au lieu de cela, l'essaim de particules explore l'espace de solutions en cherchant l'optimum global. Ainsi, Zhong et al. ont proposé l'utilisation de la matrice d'interférence pour déterminer le composant à démonter à chaque itération. Ils ont également combiné l'algorithme de Dijkstra [2]avec PSO afin de générer la meilleure SD (Zhong et al., 2011).

4.3.4 Les algorithmes basés sur l'apprentissage renforcé (RL)

L'apprentissage par renforcement est une technique spécialisée de l'apprentissage profond (You et al., 2019). Les algorithmes d'apprentissage par renforcement varient en fonction du problème étudié (Wiering & Van Otterlo, 2012). Ils peuvent être classés en deux catégories : ceux basés sur un modèle et ceux sans modèle. Les algorithmes sans modèle visent à apprendre comment fonctionne l'environnement (sa dynamique) à partir d'observations, puis à générer un modèle de comportement. Une fois qu'ils ont un modèle, ils utilisent des méthodes de planification pour trouver la meilleure stratégie. Ils sont connus pour être efficaces en termes de traitement de données, mais échouent lorsque l'espace des états possibles est trop grand. En parallèle, les algorithmes basés sur le modèle n'ont pas besoin d'apprendre l'environnement et de stocker toutes les combinaisons d'états et d'actions. Ils peuvent être divisés en deux catégories (méthodes basées sur la stratégie et méthodes basées sur la valeur), en fonction de l'objectif final de l'entraînement (Guo et al., 2023).

Dans cette section, on s'intéresse à présenter les algorithmes de Q-learning, qui sont utilisés pour l'apprentissage par renforcement (RL) et largement utilisés dans le domaine de la résolution des problèmes de désassemblage (Reveliotis, 2004, Guo et al., 2018, Chen et al., 2023). Ces algorithmes sont principalement basés sur le Processus de Décision Markovien (MPD), qui est composé de cinq éléments : l'environnement, l'agent, la ***récompense (R)***, ***l'action (a)*** et ***l'état (S)***.

(1) Tout d'abord, ***l'agent (algorithme)*** interagit avec l'environnement en effectuant une action.

(2) L'agent effectue une action et passe d'un état à un autre.

(3) Ensuite, l'agent reçoit une récompense en fonction de l'action qu'il a effectuée.

(4) En se basant sur la récompense, l'agent comprend si l'action est bonne ou mauvaise.

(5) Si l'action était bonne, c'est-à-dire si l'agent a reçu une récompense positive, alors l'agent préférera effectuer cette action. Sinon, l'agent essaiera d'autres actions qui pourraient entraîner une

[2] https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2340905

récompense positive.

Ainsi, l'apprentissage par renforcement est essentiellement un processus d'apprentissage par essais et erreurs (Ravichandiran, 2018). Le résultat de ce processus de décision, appelé ***épisode (E)***, est un scénario proposé par l'agent, qui commence par ***l'état initial*** et se termine par ***l'état cible***, en effectuant des actions pour passer d'un état intermédiaire à un autre afin de résoudre un problème proposé (Figure 9).

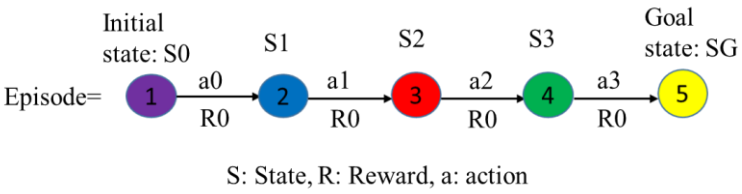


Figure 9: Exemple d'épisode

- Environnement :

L'apprentissage par renforcement (RL) est un apprentissage par essais et erreurs dans l'environnement, et l'expérience acquise est également basée sur l'environnement. Dans le domaine d'application actuel, il existe de nombreuses façons de modéliser les lignes de désassemblage dans l'environnement. Par exemple, dans le cadre du DLBP (Dismantling Line Balancing Problem), les graphes de précédence sont utilisés comme base de l'environnement dans (Tuncel et al., 2014). Les auteurs de (Xia et al., 2014) et (Zhao et al., 2021) ont adopté respectivement l'apprentissage Q-Learning et le **DQN** (Deep Q-Learning) pour apprendre les séquences de désassemblage, mais ils ont tous deux utilisé une matrice comme environnement de base pour faciliter la définition de la relation de précédence. Liu et al. (Liu et al., 2021) ont mis en place une combinaison homme-machine en utilisant l'environnement OpenAI. Ils ont adopté un mode à double agent, en prenant les opérations de la machine comme base et en considérant l'agent comme la simulation des activités des travailleurs, ce qui leur a permis d'obtenir un environnement virtuel de combinaison homme-machine.

- Etat :

Suivant les problèmes d'application, la définition des états est différente. La plupart des travaux de recherche actuels définissent les états comme une subdivision des tâches de désassemblage dans l'environnement, réalisée suivant les caractéristiques des lignes de désassemblage, mais leurs paramètres varient en fonction du problème visé. Par exemple, dans (Tuncel et al., 2014), chaque état contient des attributs tels que la station de travail et le temps d'inactivité, et un état est déterminé par la combinaison de plusieurs attributs. Dans le travail présenté dans (Xia et al., 2014), les pièces désassemblées sont considérées comme l'état, ce qui inclut les informations sur les composants, le temps de désassemblage et d'autres attributs, tout en fournissant à l'agent des informations précises et efficaces sur les pièces désassemblées, renforçant ainsi sa capacité à désassembler les produits en fin de vie.

- Action :

La capacité de sélection et de planification des actions est également un indicateur important pour évaluer les algorithmes d'apprentissage par renforcement. L'action est la concrétisation de la transition entre les états, et l'agent passe d'un état à l'autre à travers l'action. Par exemple, dans (Xia et al., 2014), l'opération de désassemblage est considérée comme une action qui est ajustée après chaque nouvel état atteint. Puis, l'ensemble des actions auxquelles chaque état est confronté, est réduit au minimum pour améliorer l'efficacité de l'exploration et la vitesse de calcul. Les auteurs de (Zhang et al., 2022) considèrent l'opération de désassemblage comme un état et la transition entre les opérations comme une action, de sorte que la matrice de l'environnement de désassemblage devient une matrice n × n qui permet de mieux exploiter ses propriétés. Dans (Zhao et al., 2021), l'action est choisie comme l'ensemble des pièces désassemblées pouvant être obtenues lorsque chaque état de désassemblage est atteint, et ces pièces désassemblées sont exprimées par une matrice relationnelle pour améliorer efficacement l'implémentation par un format standard.

- ***Récompense*** (Reward) :

Dans (Zhao et al., 2021), pendant le processus d'apprentissage, l'agent obtient des récompenses différentes suivant les différentes pièces à désassembler choisies. La récompense est la base de l'apprentissage par renforcement (RL), elle est utilisée pour fournir une rétroaction sur l'action de l'agent, et est également considérée comme une référence pour mesurer la probabilité d'atteindre un état quelconque. La récompense en RL est généralement définie en fonction de la valeur cible. Dans (Chen et al., 2019), les récompenses sont fonction des caractéristiques de minimisation du temps : un modèle d'optimisation à deux objectifs a été établi pour minimiser le temps total de réalisation maximum et la distance logistique, et la récompense a ainsi été définie comme un vecteur bidimensionnel. Dans (Liu et al., 2021), le temps de coordination homme-machine a été utilisé comme récompense dans le cas d'un environnement dynamique. Zhao et al. (Zhao et al., 2021) ont

pris le temps de désassemblage comme récompense pour l'agent en se basant sur les caractéristiques du désassemblage sélectif.

Les algorithmes de Q-learning utilisés pour le RL sont classés en deux catégories en fonction de l'environnement de l'application (Figure 10). Le réseau de neurones profond Q (**DQN**) est utilisé pour les environnements continus, tandis que le réseau Q (**QN**) est utilisé pour les environnements discrets. Les principes fondamentaux de ces deux types de réseaux sont détaillés en ***Annexe 2***. Les environnements continus sont souvent utilisés pour les animations dynamiques où l'ensemble de l'environnement 3D doit être construit pour comprendre les paramètres de DQN. Les environnements discrets dépendent principalement d'entrées numériques (matrices, dictionnaires) pour construire leur matrice de récompense (R). En termes de temps d'exécution, le QN est plus rapide que les algorithmes DQN grâce à la plus petite base de données numériques manipulée (Ravichandiran, 2020).

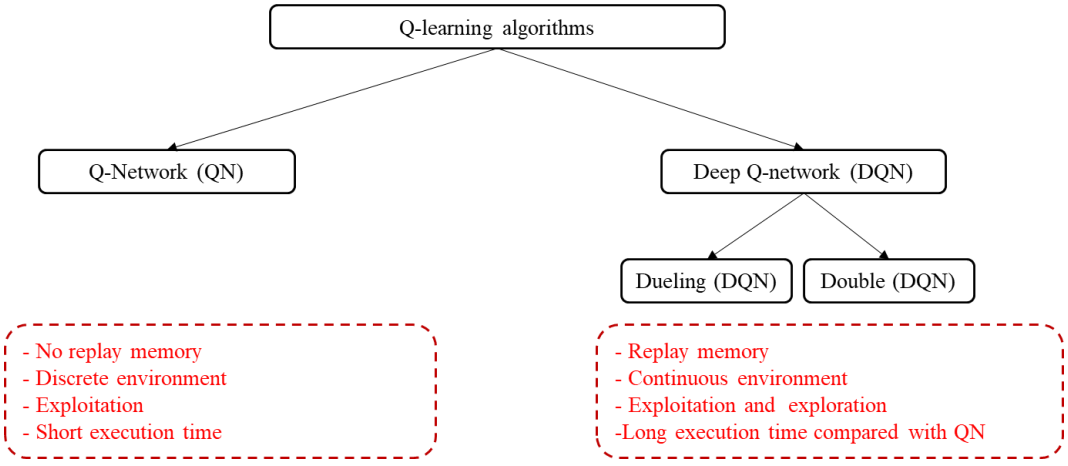


Figure 10: Algorithmes Q-learning (QN/DQN)

Il existe quelques différences clés entre QN et DQN. Q-network est un algorithme classique d'apprentissage par renforcement qui utilise une table pour stocker et mettre à jour les valeurs Q pour chaque paire état-action, tandis que DQN est une extension de Q-network qui approxime les valeurs Q à l'aide d'un réseau neuronal, ce qui lui permet de gérer des espaces d'états plus grand et continus. DQN utilise également la technique de replay d'expérience, qui contribue à stabiliser l'apprentissage et à améliorer l'efficacité des échantillons. Cependant, DQN nécessite davantage d'itérations d'entraînement et des ressources informatiques plus conséquentes par rapport à QN. Dans la résolution du problème de désassemblage, ces deux algorithmes sont largement utilisés. Le choix de l'un de ces algorithmes est basé sur les données d'entrée accessibles. Si on veut utiliser DQN pour résoudre le problème, les entrées doivent être sous la forme d'environnement continu telles que des images, des fichiers 3D etc. Il est également important de trouver le modèle de simulation adéquat qui peut prendre en charge ce type d'algorithme. L'utilisation de DQN dans les problèmes de désassemblage permet d'obtenir une résolution en animation 3D ou en vidéo, mais nécessite un temps d'exécution extrêmement long en raison de la manipulation des produits dans l'industrie (Gu et al., 2023). En revanche, QN est davantage utilisé pour manipuler des espaces d'environnement discret

tels que des matrices, des vecteurs ou des graphes. L'utilisateur n'a qu'à importer les valeurs d'entrées dans l'interface d'exécution de Q-learning (par exemple, PyTorch ou tout autre interface d'exécution Python compatible avec la bibliothèque d'intelligence artificielle).La comparaison entre ces 2 types d'algorithmes de Q-Learning est présentée dans le Tableau 1.

Tableau 1: Comparaison entre l'algorithme QN et DQN

Aspect	Q-Network	Deep Q-Network (DQN)
Approach	Table-based	Neural network-based
Function	Calculates Q-values for each state	Approximates Q-values using a neural network
State space	Discrete	Can handle both discrete and continuous states
Action space	Discrete	Discrete (can be extended to continuous)
Exploration vs Exploitation	ε-greedy exploration	ε-greedy exploration, or exploration via randomness
Memory	No memory used	Experience replay memory to store past experiences
Convergence Speed	Relatively slower convergence	Faster convergence, due to function approximation
Execution time	Generally faster	Relatively slower
Training Efficiency	Fewer training iterations	More training iterations needed

4.4 L'INTELLIGENCE ARTIFICIELLE POUR L'OPTIMISATION DES PSD

Pour profiter des avantages et performances des algorithmes d'intelligence artificielle, des chercheurs ont exploré ces méthodes pour l'optimisation des PSD. Par exemple, Li et al, ont proposé un prototype de planificateur de séquence de désassemblage pour la maintenance basé sur des graphes de contraintes, en utilisant les algorithmes génétiques (AG) (Li et al., 2005). En 2010, un nouvel algorithme Flatworm (FA) pour le problème de planification de désassemblage est proposé par Tseng et al.. Il vise à optimiser la PSD en minimisant le nombre d'outils et de changements de direction dans une séquence de désassemblage (Tseng et al., 2010). L'approche de FA a été comparée à d'autres approches visant à optimiser les mêmes paramètres AG basé sur les blocs (Tseng et al., 2018), l'optimisation par colonies de fourmis (Dorigo and Gambardella, 1997) et le système de fourmis Max-Min (Liu et al., 2012). L'analyse comparative des résultats de ces approches montre que la FA a une meilleure performance de solution et ne prend pas beaucoup de temps de calcul pour générer une PSD qui réduit le changement de direction de l'outil de désassemblage. D'autres approches ont été proposées par (Kheder et al., 2015a) pour générer un PSD optimale d'un produit à partir de son modèle CAO: la première basée sur l'AG, la seconde basée sur l'algorithme d'optimisation par colonies de fourmis (ACO). Plusieurs critères ont été introduits, tels que le volume de la pièce, le changement d'outil, les directions de désassemblage et la maintenabilité des pièces d'usure. Ensuite, une comparaison entre les algorithmes GA et ACO a été réalisée pour révéler l'efficacité de la méthode, et une étude de cas sur un frein a été mise en œuvre (Kheder et al., 2017). Le Tableau 2 représente la comparaison des auteurs entre la méthode GA et la méthode ACO sur le précédent cas d'étude pour

l'optimisation PSD afin de minimiser le changement d'outils et de direction.

Tableau 2: Analyse du résultat de la planification des séquences de désassemblage avec AG et ACO (Kheder et al., 2017)

	Nombre de changements d'outils	Nombre de changements de direction
GA	**7**	**4**
ACO	**6**	**3**

Cette comparaison montre que l'ACO est plus efficace dans la minimisation du nombre d'outils de désassemblage et du changement de direction. En utilisant ACO, Luo et al, ont proposé une nouvelle méthode pour intégrer la représentation multicouche d'un produit et la recherche de la planification optimale de la séquence de désassemblage du produit pour le désassemblage sélectif (Luo et al., 2016). En parallèle, Watanabe, K.et Inada, S. ont proposé une approche basée sur l'apprentissage par renforcement (RL) pour manipuler les bras d'un robot afin d'effectuer une séquence d'assemblage ou de désassemblage. L'apprentissage par renforcement fait partie des méthodes d'apprentissage profond et se caractérise par le concept de récompense donnée à l'agent (robot) s'il adopte le comportement attendu, et de punition s'il adopte un comportement erroné. Cette méthode a montré sa faisabilité pour des exemples simples et notamment sur le temps de désassemblage (Watanabe and Inada, 2020). De même, pour identifier le SD optimal adapté à l'incertitude structurelle des produits EoL, Zhao et al. ont développé un modèle hybride de désassemblage sélectif multi-niveaux (MSDHGM) pour illustrer les relations de contact, de priorité et de niveau entre les pièces. La PSD est ensuite formulé comme un processus de décision de Markov (**MPD**) (Zhao et al., 2021). Pour l'assemblage de produits, Zhang et al. ont utilisé avec succès le RL pour l'assemblage automatique de produits. La méthode a prouvé son efficacité pour des exemples d'assemblage simples, mais elle n'affecte pas les outils utilisés dans les opérations d'assemblage (Zhang et al., 2021).

A la lumière des travaux cité précédemment, la section suivante sert à bien montrer les besoins d'amélioration dans ce contexte de recherche. Les objectifs de la thèse seront ciblés selon les objectifs de l'industrie nouvelle.

5 ANALYSE DE L'ETAT DE L'ART ET OBJECTIFS DE RECHERCHE

Le processus d'optimisation de planification des séquences de désassemblage a été abondamment étudié dans la littérature. L'***Annexe 3*** rassemble ces principaux travaux, en précisant leur contenu et notamment, le type de modélisation/représentation utilisé, les méthodes et paramètres d'optimisation, ainsi qu'une analyse critique.

Ainsi, en examinant les recherches présentées dans ce chapitre, nous observons que la planification du PSD optimal pour un produit assemblé requiert trois étapes essentielles : la modélisation du

produit, la génération des séquences et l'optimisation des séquences identifiées. En ce qui concerne la modélisation du produit à désassembler, il existe deux approches principales. La première repose sur des méthodes basées sur les arbres et les graphes. Les avantages de ces approches peuvent être résumés comme suit :

- Les graphes ET/OU montrent une efficacité dans la détermination des différentes séquences possibles de désassemblage.
- Les Réseaux de Pétri représentent chaque sous-ensemble et chaque tâche du désassemblage.
- Le graphe d'adjacence se focalise sur les relations de priorité dans le but de générer automatiquement des séquences de désassemblage. Il présente l'avantage de produire un nombre inférieur de nœuds par rapport au graphe ET/OU.

Néanmoins, leurs inconvénients s'articulent autour de points suivants :

- Les graphes directs ou les graphes ET/OU requièrent un espace important de représentation pour un nombre de nœuds important.
- L'intégration et l'exploitation de ces graphes sont complexes, en particulier dans le cas de mécanismes complexes, ce qui entraîne une augmentation significative du temps d'exécution.
- Les relations de précédence présentant des données d'entrées pour les graphes sont introduites manuellement.

La deuxième approche de modélisation est basée sur la représentation matricielle. Ces matrices sont extraites à partir du modèle 3D CAO. Les avantages de cette approche peuvent être résumés comme suit :

- Plusieurs types de matrices (matrice d'interférence, de contact, de connexion etc…) peuvent être utilisés selon le cas d'étude.
- Facilité d'implémentation informatique.
- Adaptable pour les assemblages mécaniques avec un nombre élevé de pièces.

Néanmoins l'extraction automatique nécessite une certaine connaissance des plateformes CAO et des langages de programmation de leur interface utilisateur.

Ces différentes modélisations sont utilisées pour optimiser les séquences de désassemblage pour la maintenance (corrective ou préventive) et pour le désassemblage de fin de vie des produits mécaniques. Cette optimisation est réalisée soit par des méthodes traditionnelles variées ou des méthodes basées sur les algorithmes d'IA. Ces méthodes ont permis d'optimiser le PSD en fonction de plusieurs objectifs tels que :

- La minimisation de changements d'outils.
- La minimisation des changements de direction.
- La récupération de certaines pièces choisies.

- Minimiser les temps et les coûts de désassemblage.
- L'amélioration de l'espace de travail pour minimiser le temps de désassemblage.

En conclusion, les modèles utilisés dans la littérature pour traiter les problèmes de PSD sont principalement basés sur des méthodes de graphes ou des matrices qui traitent de divers algorithmes. En général, les approches proposées se limitent soit à la génération manuelle de SD, soit à l'optimisation d'un ou de quelques paramètres fondamentaux (outil et changement de direction) pour réduire le temps de désassemblage. Bien qu'intéressantes, elles ne prennent pas en compte certaines contraintes industrielles (accès aux pièces d'usure, minimisation du temps de préparation) qui ont une influence critique sur le temps de désassemblage. De plus, certaines méthodes d'optimisation n'exploitent pas les avantages d'algorithmes avancés plus stables qui peuvent être plus adaptés à ce type de problème de PSD en termes de temps de résolution et de quantité de paramètres pris en compte, tels que les algorithmes d'IA. En effet, selon la littérature, les algorithmes RL ont prouvé leur efficacité en termes de stabilité, de convergence plus rapide et de temps d'exécution plus court, comme l'attestent les travaux existants sur l'utilisation de ces algorithmes pour la mise en œuvre de SD déjà optimisées par des robots. Cependant, peu de travaux ont exploité les capacités de RL pour l'optimisation de PSD.

Par conséquent, nous proposons d'explorer ce domaine de recherche en s'appuyant non seulement sur les contraintes fondamentales du désassemblage, mais en prenant aussi en compte les paramètres liés aux pratiques industrielles dans le processus d'optimisation de la PSD, à travers des algorithmes plus avancés et plus stables. En effet, les travaux d'optimisation des PSD trouvés dans la littérature n'intègrent pas d'étude qui satisfasse les besoins suivants :

- La prise en compte des facteurs liés aux pratiques industrielles, tels que les temps de manipulation des assemblages etc…
- La génération des séquences de désassemblage à l'aide d'un outil qui garantisse à la fois la stabilité des résultats et la rapidité d'exécution.
- La prise en compte de la maintenance préventive lors de la génération des séquences de désassemblage.
- Améliorer la conception pour répondre aux besoins du PSD.

Sur la base des limitations constatées, l'objectif de cette thèse est de proposer une approche qui vise à optimiser les processus de planification des séquences de désassemblage partiel et de fin de vie en prenant en compte les limitations citées précédemment. Cette approche sera un support d'aide à la décision dans la phase de maintenance, de désassemblage de fin de vie et de la reconception des assemblages mécaniques. D'autres paramètres seront également considérés dans le processus d'optimisation des séquences de désassemblage tels que : la minimisation du changement de direction et du changement d'outils, l'accès aux pièces d'usure pour la maintenance et le désassemblage des pièces de petites tailles en premier. Enfin, ces résultats seront alors exploités pour proposer des alternatives de reconception des architectures CAO des assemblages mécaniques afin d'optimiser les séquences de désassemblage en minimisant non seulement le temps de désassemblage mais aussi le changement de direction et d'outils utilisés.

Les principaux objectifs de cette thèse ont été représentés sur la Figure 11.

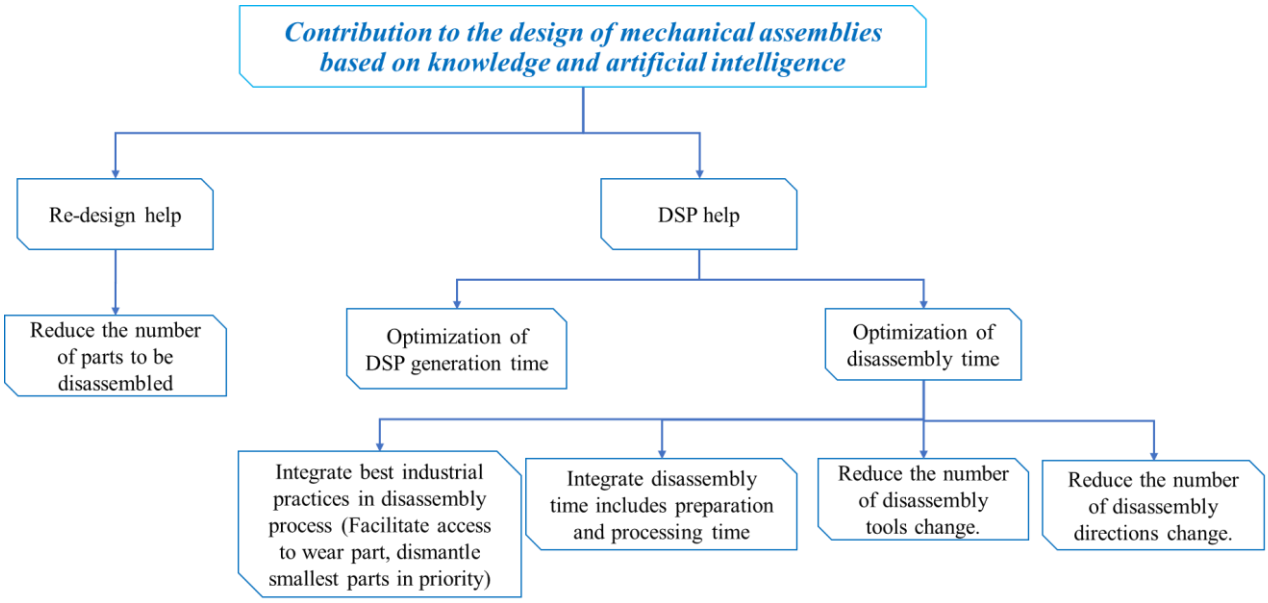


Figure 11: Synthèse des objectifs de la thèse identifiés

Il s'agira donc de :

- Développer un outil intelligent pour l'optimisation de la planification des séquences de désassemblage afin de :
  - Minimiser le temps de réalisation de séquence de désassemblage.
  - Minimiser le changement des directions lors des opérations de désassemblage.
  - Minimiser le changement des outils lors des opérations de désassemblage.
  - Minimiser le temps d'exécution algorithmique pour fournir des résultats dans le minimum de temps possible.
- Proposer une méthodologie de reconception des assemblages mécaniques permettant d'avoir des séquences de désassemblage partiel optimales.

Le périmètre des verrous adressés dans cette thèse au regard des exigences/contraintes présentées de la Figure 5 couvre l'ensemble des exigences, exceptée celle de l'assemblage parallèle.

Ainsi, la contribution de cette thèse va intervenir après la phase d'utilisation du produit : une première contribution visera le désassemblage partiel pendant la phase de maintenance mais aussi le désassemblage total pour la phase de fin de vie. Une deuxième contribution adressera la phase de reconception pour améliorer la conception de l'assemblage afin d'en faciliter le désassemblage.

6 CONCLUSION

Dans ce chapitre, nous avons présenté les travaux de la littérature visant à améliorer le processus de désassemblage de produits. Nous avons commencé par examiner les différentes techniques de modélisation des assemblages mécaniques. Pour chaque approche, nous avons présenté le principe général ainsi que les avantages et les limites associés. Ensuite, nous avons abordé les méthodes de génération des séquences de désassemblage et les méthodes d'optimisation existantes pour résoudre ces problèmes. Cette analyse nous a permis de dresser une synthèse des avantages et des inconvénients de ces techniques. Ces conclusions nous ont aidés à définir les objectifs de la thèse et à établir la structure de ce mémoire.

Dans le prochain chapitre, nous présenterons notre première contribution concernant la méthode d'optimisation des PSD par RL.

CHAPITRE 2 : OPTIMISATION DES SEQUENCES DE DESASSEMBLAGE PAR L'APPRENTISSAGE PAR RENFORCEMENT

1 INTRODUCTION

Actuellement, les produits mécaniques et mécatroniques ont une structure interne variée qui rend difficile la modélisation et le développement de plans de désassemblage efficaces. Les chercheurs utilisent plusieurs algorithmes pour obtenir les séquences de désassemblage (SD) optimales. Ces approches rencontrent souvent des difficultés de réalisations pratiques. Par conséquent, de nombreuses nouvelles études commencent à adopter des méthodes d'optimisation basées sur des algorithmes d'intelligence artificielle et notamment d'apprentissage profond nommé Deep Learning (DL) pour résoudre les problèmes de planification des séquences de désassemblage (PSD). Parmi les algorithmes DL, l'apprentissage par renforcement nommé Reinforcement Learning (RL) est le plus adapté aux problèmes de prise de décision. Il détermine le comportement optimal en interagissant avec l'environnement et en observant ses réactions pour proposer les meilleures décisions. Ces algorithmes sont également plus efficaces en termes de temps d'exécution et de stabilité des résultats, même s'ils nécessitent d'être adaptés aux pratiques industrielles du désassemblage.

Ce chapitre décrit une première contribution pour optimiser les SD sur les plans à la fois technique et méthodologique.

La contribution technique concerne l'optimisation du temps de la SD. Basée sur l'algorithme RL, la réduction du temps de réalisation des SD intègre cinq paramètres de désassemblage à savoir : la minimisation des changements d'outils de désassemblage, la minimisation des changements de direction de désassemblage, l'optimisation du temps de démontage, y compris le temps de préparation et de traitement et enfin la priorité de démontage des plus petites pièces et la facilité d'accès aux pièces d'usure.

La contribution méthodologique se concentre sur l'intégration d'un algorithme d'optimisation basé sur l'apprentissage par renforcement dans un processus de désassemblage, en commençant par les données CAO jusqu'à la génération de la SD optimale selon les priorités prédéfinies. Cette méthodologie a été mise en œuvre en développant un outil en langage de programmation Python pour valider l'approche proposée.

Dans ce chapitre, nous allons décrire la méthode d'optimisation par l'algorithme d'apprentissage par renforcement (Q-Learning) pour la génération des SD. Dans un premier temps, les hypothèses de l'approche développée seront présentées. Puis, les différentes étapes de l'algorithme développé seront détaillées, en se basant sur les données, extraites automatiquement de la CAO d'un produit. Un exemple illustratif d'un mécanisme de guidage par roulement sera traité tout au long de cette présentation afin de mieux expliquer les différentes étapes de la démarche proposée.

2 APPROCHE PROPOSEE

L'approche proposée porte sur la génération automatique de la SD optimale pour la maintenance et la fin de vie en utilisant un algorithme d'apprentissage par renforcement (RL), et plus spécifiquement l'algorithme du réseau Q (QN). L'algorithme QN est l'un des algorithmes basés sur les Processus de Décision de Markov (PDM). Les PDM sont une manière mathématique de formaliser un problème de décision séquentielle. Cette contribution est basée sur les hypothèses suivantes :

- **Hypothèse 1 :** Le démontage des pièces est réalisé d'une façon séquentielle. Cela signifie qu'on devrait désassembler pièce par pièce ou sous-ensemble par sous-ensemble. Le désassemblage de deux pièces en parallèle n'est pas pris en compte dans le cadre de cette thèse.

- **Hypothèse 2 :** Les contacts entre les pièces sont supposés parfaits et les pièces mécaniques sont supposées rigides.

- **Hypothèse 3 :** Le désassemblage est supposé statique, c'est-à-dire que le mouvement des pièces ne sera pas fonction du temps. Pour un désassemblage de (n) pièces, nous passons directement de (n) pièces à (n-1) pièces sans représenter les détails des mouvements dynamiques.

- **Hypothèse 4 :** Chaque pièce mécanique ne peut être démontée qu'avec un seul outil. Cet outil ne change pas selon l'ordre de démontage de la pièce dans la séquence.

La Figure 12 présente les principales étapes proposées pour la génération des SD pour le désassemblage partiel ou total (la maintenance ou la fin de vie). La première étape est le traitement des données issues de la CAO. Une fois le traitement des données terminé, l'algorithme commence à remplir la matrice de récompense qui représente les états et les actions dans l'environnement étudié. La matrice de récompense est l'entrée principale des algorithmes d'apprentissage par renforcement. L'algorithme adopté dans cette approche est le QN. L'algorithme QN chargé de la création des épisodes de démontage du système mécanique étudié.

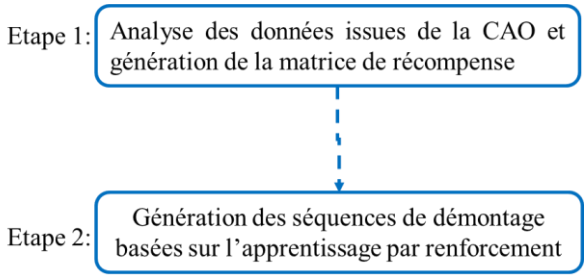


Figure 12: Les deux étapes principales de l'approche proposée

Les données CAO traitées sont extraites à partir du modèle CAO du système étudié (depuis l'environnement Solidworks©). L'extraction de ces données peut se faire dans de nombreuses extensions de fichier tels qu'un tableau Excel, un fichier texte ou XML (Hadj et al., 2018a). Une fois

les données CAO extraites, elles serviront de base à l'approche proposée, comme entrées initiales, au travers de la matrice de collision (MC), les noms et numéros des pièces ainsi que les outils de démontage, les volumes des pièces à démonter, en plus des temps standards de démontage et de préparation pour le démontage.

Concernant la MC, dans les logiciels de conception assistée par ordinateur (CAO), on distingue deux catégories de collisions : les collisions statiques et les collisions dynamiques. Les collisions statiques sont automatiquement repérées lorsqu'il y a une collision entre deux éléments. Par exemple, lors de l'assemblage d'un arbre de 100 mm de diamètre avec un alésage de 50 mm de diamètre, le logiciel détectera une collision statique entre ces deux éléments. Pour ces collisions statiques, le concepteur peut interroger directement le système de CAO pour obtenir cette information en temps réel à l'aide d'une commande explicite.

En revanche, les collisions dynamiques dans un assemblage ne sont détectées que lorsqu'une pièce est déplacée le long d'une direction (k) avec un certain pas d'incrément, et qu'une collision fictive est détectée. Ce test permet d'identifier les interférences de chaque pièce avec les autres pièces de l'assemblage en les déplaçant le long d'une direction spécifique. Pendant le test de collision, les mouvements testés concernent les déplacements le long des 3 axes x, y et z. Les résultats de ces tests sont représentés par trois matrices distinctes. Ces matrices, notées [MIx], [MIy] et [MIz], sont des matrices carrées de taille (n×n). L'élément MIk(i,j) de la matrice indique si une collision existe ou non entre la pièce (i) et la pièce (j) lorsque la pièce (i) est déplacée dans la direction (k). Plus précisément :

- MIk (i,j) = 1 s'il y a une collision entre la pièce (i) et la pièce (j) lors du déplacement de la pièce (i) selon la direction (k), où k peut être l'un des axes : (x), (y) ou (z).
- MIk (i,j) = 0 s'il n'y a pas de collision entre la pièce (i) et la pièce (j) lors du déplacement.
- MIk (i,j) = 0 si i = j, c'est-à-dire lorsque la pièce (i) est comparée à elle-même, dans ce cas, il n'y a pas de collision.

La matrice de collision est utile pour la planification des SD pour la maintenance et la fin de vie. Elle donne toutes les interactions entre les pièces dans les trois directions dans le référence cartésien (X, Y, Z). Elle constituera le moteur pour la construction de la matrice de récompense. Tandis que la matrice de récompense servira de « guide » pour l'agent QN afin de choisir la meilleure décision. Enfin, cet agent QN sera le responsable de l'extraction des SD. La section suivante présente la première étape de l'approche proposée, à savoir l'analyse des données qui proviennent de la CAO.

3 ANALYSE DES DONNEES CAO

Au sein d'un modèle CAO, chaque composant est caractérisé par différents attributs tels que la direction de désassemblage, le chemin d'accès, l'inertie, la masse, le volume, le nombre des relations et les contraintes d'assemblage. Toutes ces informations sont obtenues à partir de l'environnement

CAO. Dans notre équipe de recherche, nous avons développé un outil d'extraction automatique des données CAO appelé CADLAB©. Cet outil permet d'accéder aux informations spécifiques d'un assemblage mécanique dans le but de les convertir en données exploitables pour l'assemblage/désassemblage. Dans notre étude, nous allons utiliser cet outil pour extraire les données CAO requises (Hadj et al., 2018a). Afin d'illustrer notre démarche nous introduisons un exemple illustratif, celui d'un guidage en rotation.

- Présentation de l'exemple illustratif

L'exemple choisi (Figure 13) est un mécanisme de guidage en rotation de 5 pièces pour diriger et contrôler le mouvement de rotation de l'arbre (3) autour d'un axe fixe (x). Cet exemple est constitué de deux roulements de type BC (2,5), une bague entretoise (4), un arbre (3) et un moyeu (1) (Tableau 3). La première contribution de la thèse sera illustrée étape par étape en s'appuyant sur cet exemple.

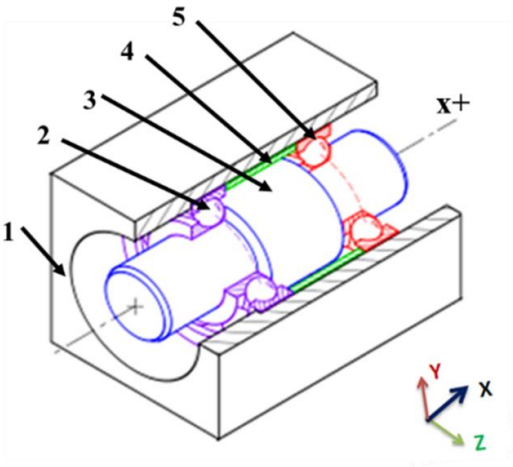


Figure 13: Exemple illustratif

Tableau 3: Liste des pièces de système de guidage en rotation

Pièce	Numéro	Nom	Outil de désassemblage
1	1	Moyeu	Presse hydraulique
2, 5	2	Roulement	Extracteur de roulement
3	1	Arbre	Presse hydraulique
4	1	Bague entretoise	Démontage à la presse / manuel

Le détail de l'algorithme des activités d'analyse des données CAO est donné sur la Figure 14.

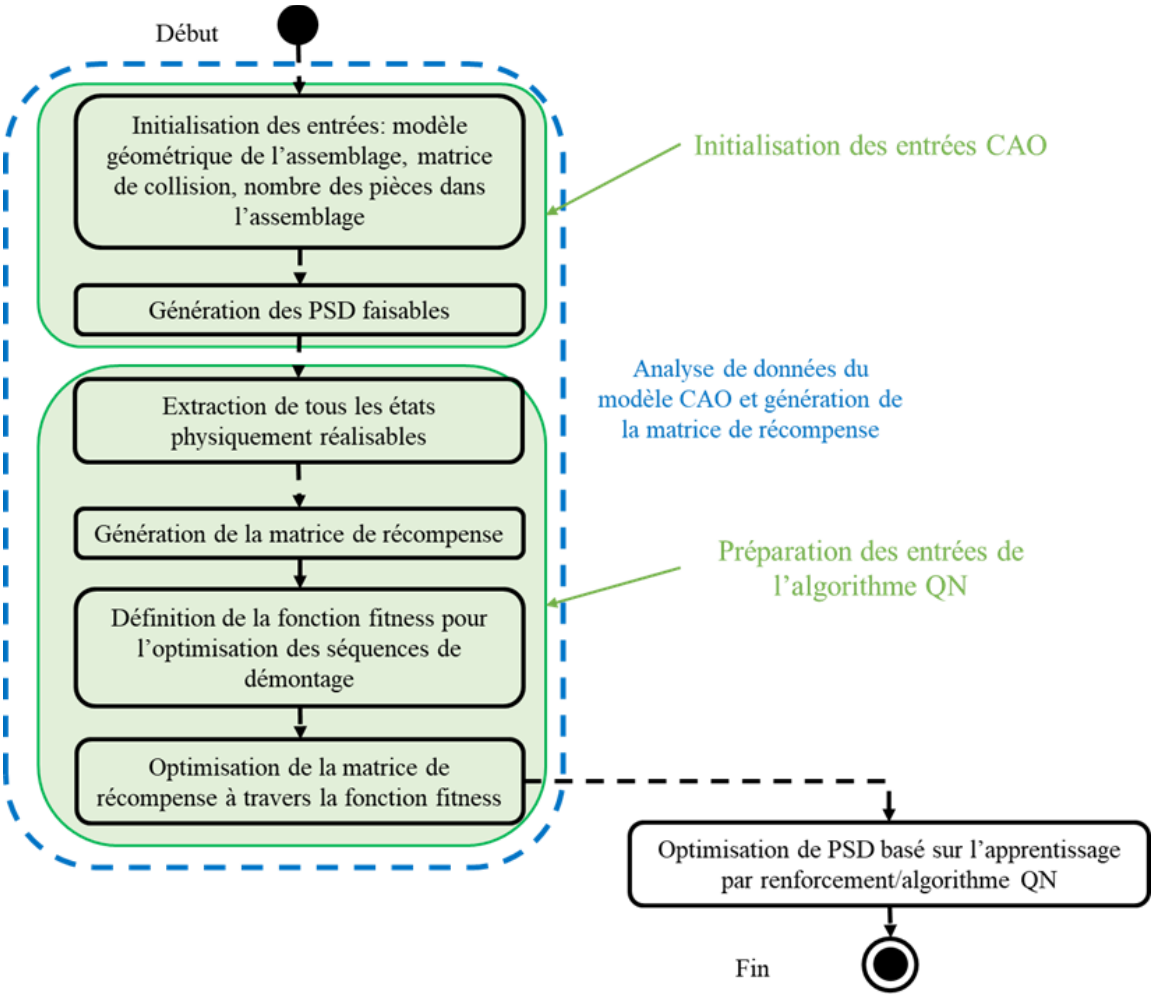


Figure 14: Algorithme d'analyse des données CAO

3.1 INITIALISATION DES ENTREES GEOMETRIQUES

- Initialisation des données CAO

La première étape de l'approche commence par l'analyse des données extraites du modèle CAO. Les entrées de l'algorithme d'analyse de données sont : la ou les matrices de collision suivant que le démontage est uni ou multidirectionnel ainsi que le nombre de pièces dans l'assemblage.

- Génération des SD faisables à partir de la MC

La Figure 15 montre comment extraire une SD à partir de la matrice de collision. Pour extraire une SD dans la direction (+x), la première étape consiste à identifier la ou les lignes de la matrice de collision dont les valeurs cumulées sont égales à zéro (c'est-à-dire les pièces sans collision). Les pièces correspondantes, ici la pièce 5, sera retirée et placée dans la SD ; la ligne et la colonne correspondantes dans la matrice de collision doivent alors être supprimées (ligne et colonne 5). L'étape 2 représente la nouvelle matrice de collision obtenue lorsque les informations correspondantes

aux pièces supprimées (Pièce 5) sont effacées. Ensuite, le processus est similaire avec les pièces (3) et (4) dans la deuxième étape, et enfin les pièces (2) et (1) dans la dernière étape. Cette méthode donne une SD dans la direction (+x). Pour la PSD dans la direction (-x), nous répétons le même processus mais avec la somme des colonnes de la matrice de collision. Dans le cas d'exemples plus complexes avec des directions de désassemblage en 3D et un grand nombre de pièces, ce traitement prendrait plus de temps s'il était effectué manuellement. Par conséquent, nous avons automatisé toutes ces opérations avec un script Python.

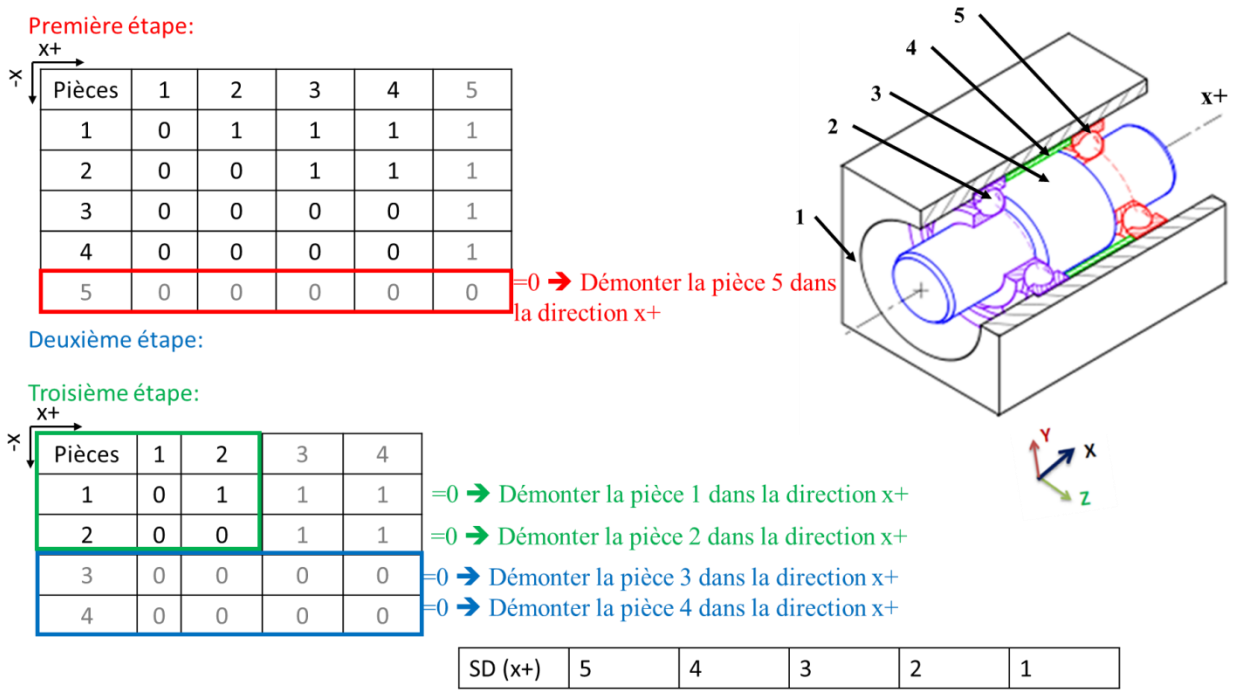


Figure 15: Description de la génération de la SD à l'aide de la matrice de collision.

En tenant compte des multiples possibilités de démontage à chaque étape (lorsque plus d'une ligne/colonne ont une somme égale à zéro), pour l'exemple choisi de cinq pièces, on identifie 22 SD réalisables (**Error! Reference source not found.** p.**Error! Bookmark not defined.**).

3.2 PREPARATION DES ENTREES DE L'ALGORITHME QN

Comme précisé dans l'étude bibliographique l'algorithme QN a son propre vocabulaire à définir pour chaque type de problème tel que l'environnement sur lequel nous allons travailler. QN définit l'environnement comme le monde de l'agent. Ici, l'environnement est le produit assemblé ou le sous-assemblage mécanique ou mécatronique et l'agent est l'algorithme QN. Nous devons également définir les entrées de l'environnement telles que : le nombre de pièces dans l'assemblage (n), la matrice de collision de l'assemblage qui est soit la matrice de collision complète (CCM) pour le désassemblage complet, soit la matrice de collision partielle (PCM) pour le désassemblage partiel. La deuxième étape consiste à intégrer le modèle mathématique pour identifier le nombre d'états

logiques (S), le nombre d'actions (A) et le nombre d'épisodes (E) pour atteindre l'état cible. L'état est une position dans l'environnement dans laquelle l'agent peut rester, et il peut y avoir plusieurs positions dans l'environnement dans lesquelles l'agent peut rester. Ces positions sont appelées des états. L'état est généralement nommé (S) pour « State » en anglais. Les passages entre les états est effectué selon la valeur de récompense donnée. La récompense nommée (R) pour « Reward » en anglais.

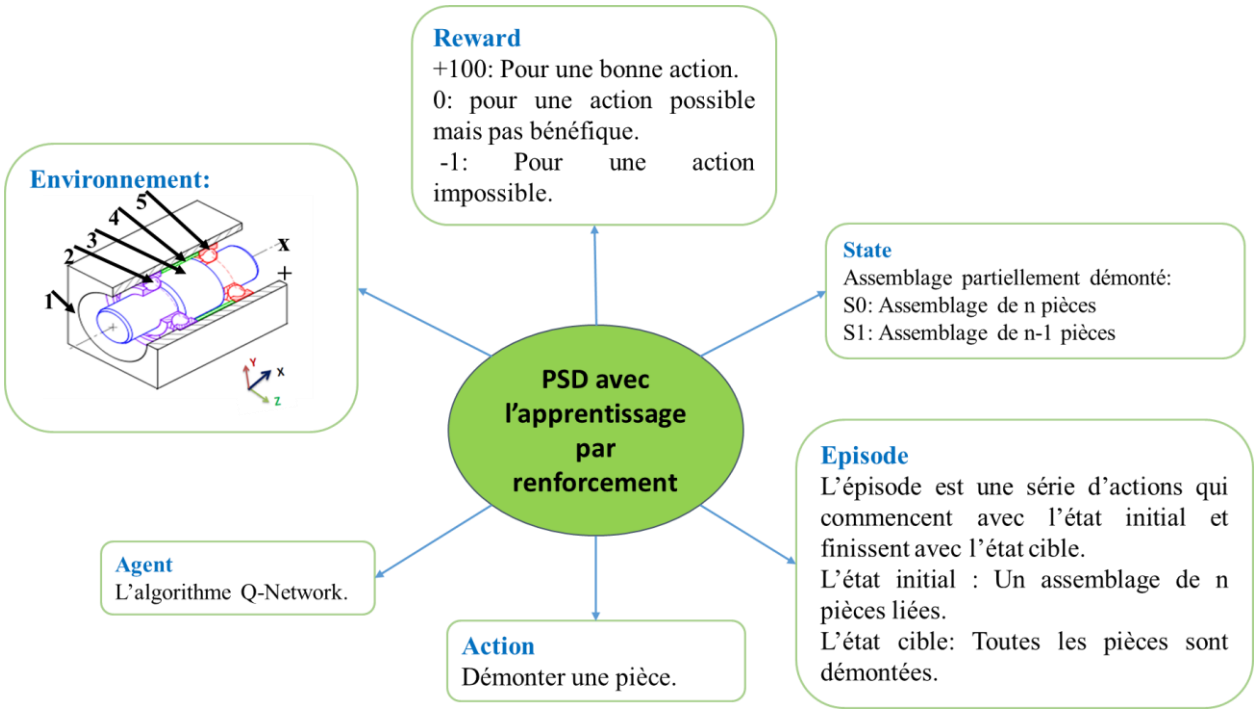


Figure 16: Les composants du réseau QN pour la planification des SD

- **Extraction de tous les états physiquement possibles**

Le processus de génération des Etats Physiquement Réalisables (EPRs) débute par l'état initial (S0), c'est-à-dire l'assemblage complet avec toutes les pièces. Pour l'assemblage de la Figure 13, S0=(1,2,3,4,5). Pour passer d'un état à un autre, l'algorithme doit effectuer une action, qui consiste à retirer une pièce. Ainsi, l'état suivant représente l'assemblage sans cette pièce. Cette action doit obéir à une loi physique, qui représente ici le processus d'extraction d'une SD à travers la matrice de collision. Le processus de génération de l'EPR est représenté dans la Figure 17.

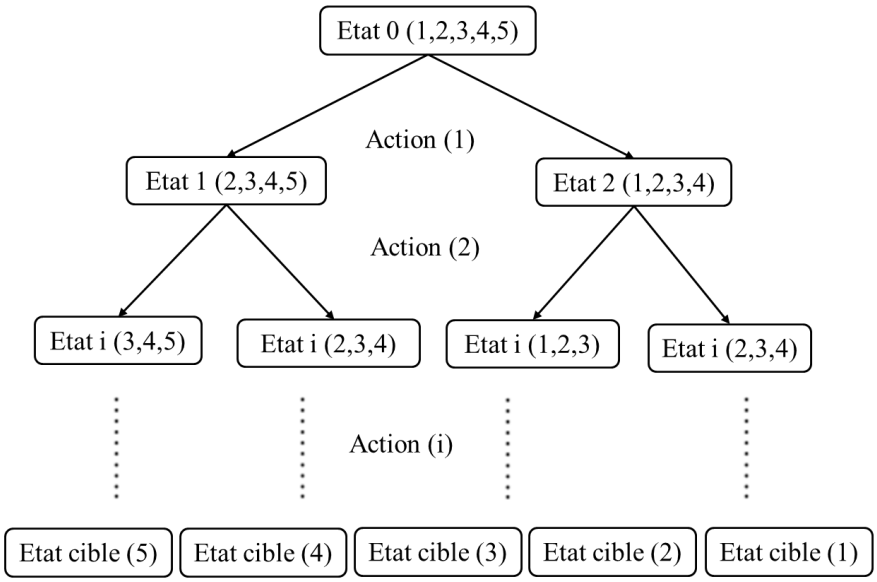


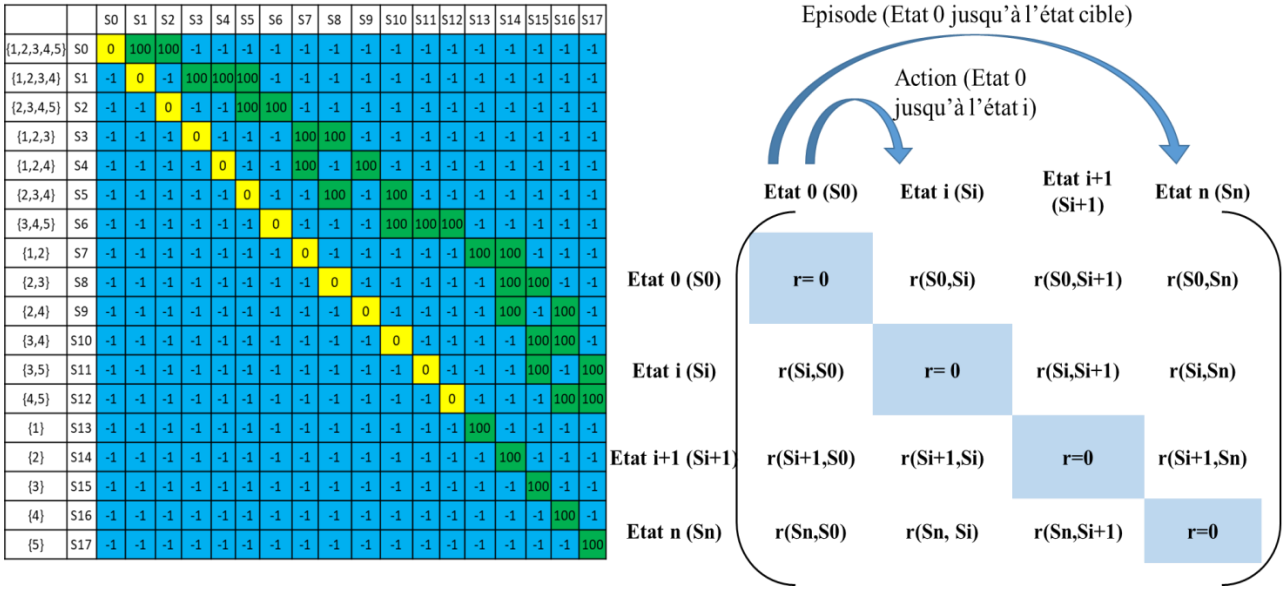
Figure 17: Processus de génération des états physiquement réalisables

L'état final dépend du type de désassemblage. Pour le processus de désassemblage en FV, l'état final correspond à l'un des cinq états finaux. Pour la Maintenance Préventive (MP), l'état final peut être l'un des états de S2 à S (final - 1) correspondant au composant d'usure à démonter.

- **La génération de la matrice de récompense basée sur l'EPRs**

Une fois la génération des EPRs terminée, l'algorithme commence par remplir la matrice de récompense qui représente les états/actions dans l'environnement étudié. La matrice de récompense (R) est une matrice carrée d'états-actions qui représente les récompenses et les punitions pour passer d'un état à un autre par le biais d'actions. Le schéma de traitement des données présentées dans la Figure 18 décrit les étapes de remplissage de la R. Le processus de récompense r de désassemblage pour passer d'un état S_i à un état S_{i+1} est le suivant :

- r (S_i , S_{i+1}) = 100 pour une bonne action : retirer une pièce ou arriver à un état final.
- r (S_i , S_{i+1}) = 0 pour une action possible sans bénéfice.
- r (S_i , S_{i+1}) = -1 pour les actions physiquement impossibles : retirer une pièce qui ne peut pas être physiquement retirée.



		S0	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	S16	S17
{1,2,3,4,5}	S0	0	100	100	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
{1,2,3,4}	S1	-1	0	-1	100	100	100	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
{2,3,4,5}	S2	-1	-1	0	-1	-1	100	100	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
{1,2,3}	S3	-1	-1	-1	0	-1	-1	-1	100	100	-1	-1	-1	-1	-1	-1	-1	-1	-1
{1,2,4}	S4	-1	-1	-1	-1	0	-1	-1	100	-1	100	-1	-1	-1	-1	-1	-1	-1	-1
{2,3,4}	S5	-1	-1	-1	-1	-1	0	-1	-1	100	-1	100	-1	-1	-1	-1	-1	-1	-1
{3,4,5}	S6	-1	-1	-1	-1	-1	-1	0	-1	-1	-1	100	100	100	-1	-1	-1	-1	-1
{1,2}	S7	-1	-1	-1	-1	-1	-1	-1	0	-1	-1	-1	-1	-1	100	100	-1	-1	-1
{2,3}	S8	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	-1	-1	-1	-1	100	100	-1	-1
{2,4}	S9	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	-1	-1	-1	100	-1	100	-1
{3,4}	S10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	-1	-1	-1	100	100	-1
{3,5}	S11	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	-1	-1	100	-1	100
{4,5}	S12	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	-1	-1	100	100
{1}	S13	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	100	-1	-1	-1	-1
{2}	S14	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	100	-1	-1	-1
{3}	S15	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	100	-1	-1
{4}	S16	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	100	-1
{5}	S17	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	100

Figure 18: Processus de remplissage de la matrice de récompense (R)

La matrice de récompense est automatiquement remplie, grâce à un script Python. La première étape consiste à initialiser une matrice avec la taille des états, calculée à partir du nombre d'états réalisables. Cette matrice initiale est automatiquement remplie avec des -1, exceptée la diagonale qui est remplie de zéros. Ensuite, la combinaison des états compatibles avec le désassemblage extrait du traitement de la matrice de collision sont remplis avec des 100. Une fois que la matrice de récompense est remplie, une fonction fitness doit être définie en se basant sur la fonction objectif qui spécifie les paramètres d'optimisation prédéfinis.

- **Définition de la fonction fitness à maximiser**

La Fonction Fitness (FF) qu'on cherche à maximiser, donnée par l'équation (1) est une équation mathématique qui englobe tous les paramètres d'optimisation suivants :

- Le temps de désassemblage,
- Les pratiques industrielles qui englobent deux paramètres : le volume relatif de la pièce et un facteur qui spécifie si la pièce à désassembler est une pièce d'usure ou non,
- Le changement de direction entre deux pièces désassemblées consécutives.

Ces paramètres sont détaillés ci-dessous.

Pour une pièce (i), la FF est exprimée par l'équation (1). Dans la FF, l'état est une étape temporairement stable, où il n'y a pas d'opération de désassemblage en cours. L'action est l'opération de désassemblage effectuée pour passer d'un état à un autre.

Équation 1:

$$F(etat, action) = \alpha \times \left(1 - \left(\frac{V_{(i)}}{Vmax}\right)\right) + \beta \times \left(1 - \left(\frac{t_{(i)}}{t_{max}}\right)\right) + \gamma \times D(i) + \delta \times T(i) + \lambda P(i)$$

Où :

- Vmax est le volume de la plus grande pièce de l'assemblage ;
- Vi est le volume de la i^{ème} pièce à retirer ;
- Pi est un booléen qui spécifie s'il s'agit d'une pièce d'usure ou non, les scores attribués sont les suivants :
  - → 1 s'il s'agit d'une pièce d'usure à retirer ;
  - → 0 si ce n'est pas une pièce d'usure ;
- t(i) est le temps nécessaire pour désassembler une pièce (i). Il comprend le temps total de préparation de la pièce ou du sous-ensemble (tprep) et le temps total de désassemblage de la pièce ou du sous-ensemble (td) tel que t(i) = tprep(i) + td(i) ;
- tmax = Max(t(i)) pour l'assemblage considéré ;
- D(i) représente l'évaluation du changement de direction lors du désassemblage, les scores attribués sont les suivants :
  - → 1 s'il n'y a pas de changement de direction entre deux pièces consécutives ;
  - → 0,5 s'il y a un changement de direction de 90° entre deux pièces consécutives (par exemple : (x^y) = 90°) ;
  - → 0 s'il y a un changement de direction de 180° entre deux pièces consécutives (par exemple : (x^-x) = 180°) ;
- T(i) est un booléen qui représente le changement d'outil, et vaut :
  - → 1 s'il n'y a pas de changement d'outil entre deux pièces consécutives ;
  - → 0 s'il y a un changement d'outil entre deux pièces consécutives ;
- α, β, γ, δ et λ représentent les coefficients de pondération avec α + β + γ + δ + λ = 1.

- **Optimisation de la matrice de récompense à travers la fonction fitness**

Une fois que la FF est définie et mise en forme de vecteur, elle est appliquée à la matrice de récompense (Figure 19). Ce vecteur « fitness » F (état, action) contient la valeur de chaque poids appliqué à la récompense du passage d'un état à un autre. La nouvelle valeur de récompense (r') est celle la valeur de récompense (r) multipliée par la valeur de fitness F(état, action) (Équation 2). Cette pondération permettra de favoriser les passages entre les états suivant les paramètres d'optimisation de désassemblage introduits dans l'Équation 2.

Équation 2 :

$$r'(S_i, S_{i+1}) = F(état, action) \times r(S_i, S_{i+1})$$

Pour minimiser le temps de traitement, le vecteur F ne sera traité que dans les cas où :

$r(S_i, S_{i+1}) = 100.$

R'=		S0	Si	Sn
	S0	r= null	r'(S0,Si)	r'(S0,Sn)
	Si	r'(Si,S0)	r= null	r'(Si,Sn)
	Sn	r'(Sn,S0)	r'(Sn,Si)	r=null

Figure 19: La matrice de récompense pondérée

Après avoir pondéré la matrice de récompense, cette matrice est intégrée dans un algorithme QN (Q-Network) pour fournir la ou les SD optimisée(s). Cela est détaillé dans la section suivante.

4 OPTIMISATION DES SD BASEE SUR L'ALGORITHME QN

L'algorithme QN repose sur une approche d'apprentissage par renforcement qui implique des essais et des erreurs dans l'interaction avec un environnement. Les composants clés des approches d'apprentissage par renforcement comprennent l'environnement, l'agent, l'état, l'action, la récompense et l'épisode. Dans cette étude, l'environnement est représenté par un assemblage mécanique 3D, avec l'algorithme QN agissant en tant qu'agent. L'agent effectue des actions basées sur la récompense pour progresser d'un état à un autre jusqu'à atteindre l'état final, et cette séquence est dénommée "épisode". Le processus d'apprentissage par renforcement repose sur la fonction de qualité Q, formulée dans une équation suivante :

Équation 3 :

$$Q[état_{courant}\ S][action\_possible\ A] = récompense + (\gamma \times (QMax[état\_suivant]))$$La fonction Q est souvent appelée fonction de qualité, évalue la qualité d'une action donnée dans un état particulier, en prenant en compte la récompense attendue et les actions futures. Formellement, la fonction Q est notée Q (s, a), où "s" représente l'état courant et "a" l'action possible. Ainsi, la fonction Q évolue au fil du temps à mesure que l'agent apprend à prendre des décisions optimales dans son environnement pour maximiser les récompenses à long terme.

Le facteur d'actualisation γ détermine l'importance des récompenses futures.

Si γ = 0 rendrait l'agent myope ou à courte vue.

Si γ = 1 introduirait le risque de divergence des récompenses plus éloignées dans la fonction Q.

- **Initialisation des paramètres de QN**

Les paramètres du QN sont : le nombre de pièces dans l'assemblage, un tableau vide nommé Q-table de la même taille que la matrice R. Ce dernier sera rempli ultérieurement avec les valeurs de Q selon l'équation Q, l'état final, le facteur d'actualisation γ, la matrice de récompense R' pondérée et l'architecture de l'assemblage. Dans l'exemple illustratif de la Figure 13, pour un désassemblage FV:

- La matrice R est déjà calculée par l'algorithme de traitement de la MC.
- La taille de la table Q vide est la même que celle de la matrice R, soit [18, 18] dans notre exemple (Figure 18).
- L'état initial est l'état (S0) où toutes les pièces sont assemblées.
- L'état final est l'un des états : S13, S14, S15, S16, S17 où il n'y a aucune pièce liée (désassemblage FV).

Le facteur d'actualisation γ est de 0,8. Cette valeur a été choisie après une étude paramétrique en variant la valeur de γ et en regardant l'effet sur les résultats trouvés. Si γ tend vers 0, l'agent sera myope ou à courte vue et ne voit que les récompenses les plus proches. γ = 1 introduirait le risque de divergence des récompenses plus éloignées dans la fonction Q car l'agent ne prendrait en compte que les valeurs de récompenses les plus lointaines.

Les différentes activités de l'algorithme d'apprentissage par renforcement, implémentées par des scripts en Python sont présentées sur la Figure 20.

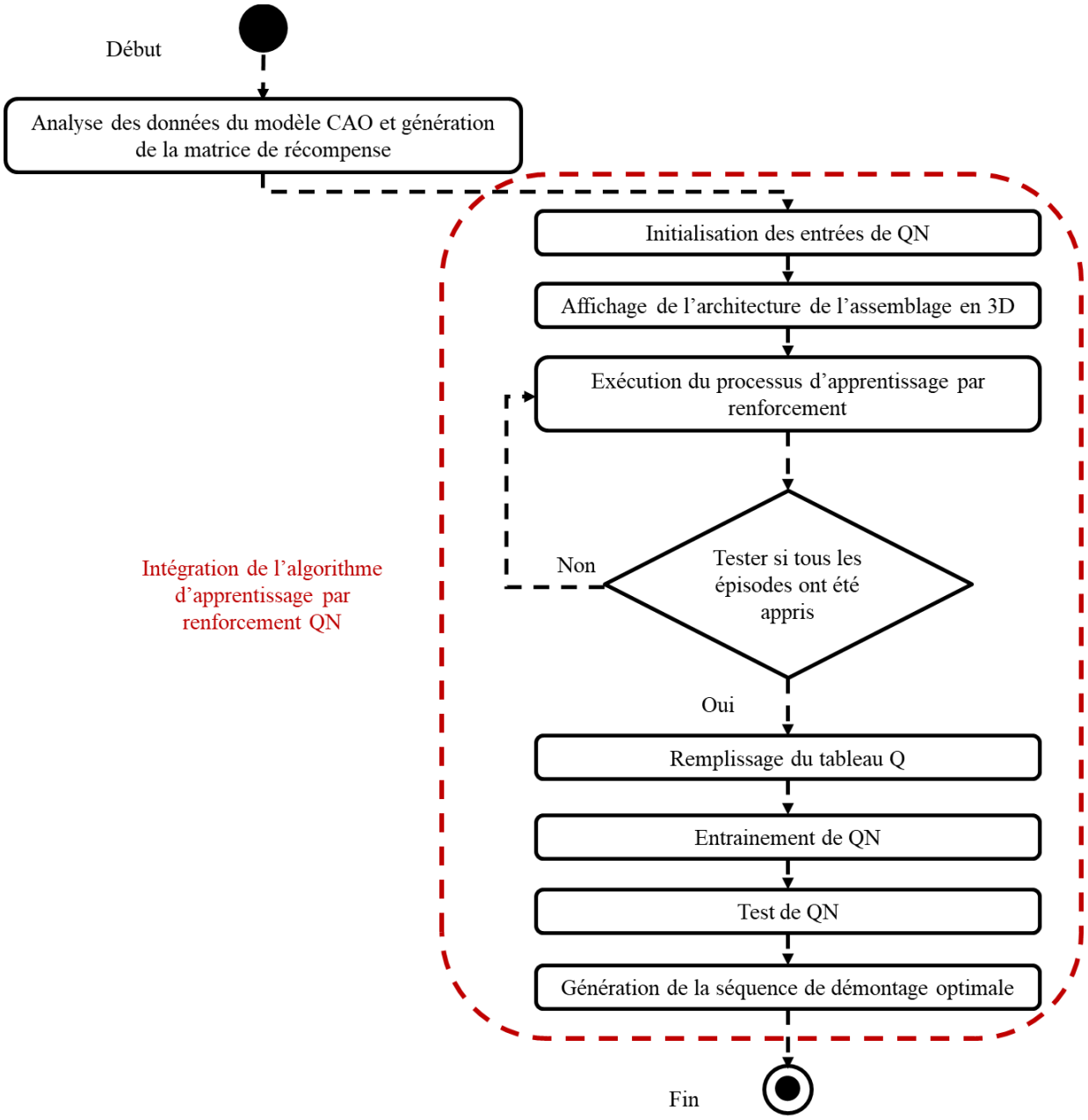


Figure 20: Optimisation de la PSD basée sur l'algorithme QN implémentée en Python

- **Affichage de l'architecture de l'assemblage**

Le script affiche l'architecture de l'assemblage dans l'interface utilisateur en 3D.

- **Apprentissage RL**

Après avoir exécuté l'équation Q de l'algorithme QN sur tous les épisodes possibles, l'agent sera en mesure de construire la table Q et d'identifier les décisions de désassemblage en se basant sur les valeurs maximales de Q.

- **Remplissage du tableau Q**

Répétez l'exécution de l'équation Q jusqu'à ce que la table Q soit remplie avec tous les épisodes. La table Q contient les valeurs de Q calculées en fonction de l'équation Q (S, A) (Équation 3). Une fois que la table Q est prête, l'agent QN commence à créer tous les épisodes de désassemblage (SD) possibles du produit assemblé mécanique ou mécatronique en choisissant les valeurs Q maximales pour passer d'un état à un autre.

- **Génération d'une PSD optimale**

Une fois que la table Q est remplie, l'algorithme commence à rechercher l'épisode (SD) de désassemblage optimal avec le score le plus élevé dans la table Q. Ensuite, l'algorithme QN affiche l'épisode de désassemblage optimal. Dans la section suivante, l'approche proposée est illustrée sur l'exemple de cinq pièces (Figure 13).

4.1 APPLICATION SUR L'EXEMPLE ILLUSTRATIF

Cette section détaillera l'application de l'approche développée sur l'exemple représenté dans la Figure 13. Les informations de l'assemblage sont représentées dans le Tableau 3 p.42.

4.1.1 Collecte des données CAO

Pour initialiser les entrées de l'algorithme QN, les données CAO comprennent d'une part la matrice de collision qui décrit les risques de collision entre les pièces selon les trois directions (X, Y, Z), et d'autre part un tableau d'informations sur l'assemblage. Ce tableau contient tous les détails permettant de générer les SD tels que le temps de désassemblage de chaque pièce, son volume, son outil de désassemblage et spécifie les pièces d'usure dans l'assemblage pour les besoins de maintenance préventive. Le Tableau 4 illustre ces informations.

Tableau 4: Données de l'assemblage relatives à l'exemple illustratif

Pièce (numéro(s))	Roulement (2,5)	Moyeu(1)	Arbre(3)	Bague d'entretoise (4)
Volume (mm^3)	39307	341862	675442	74612
Temps de désassemblage unitaire (s)	45	35	15	15
Outil	Extracteur de roulement	Presse hydraulique	Presse hydraulique	Main/ Presse
Pièces d'usure	Oui	Non	Non	Non

La matrice de collision de l'exemple à 5 pièces de la Figure 13 est la suivante :

$$[CM_{(X,Y,Z)}] = \begin{bmatrix} 000 & 100 & 100 & 100 & 100 \\ 000 & 000 & 100 & 100 & 100 \\ 000 & 000 & 000 & 000 & 100 \\ 000 & 000 & 000 & 000 & 100 \\ 000 & 000 & 000 & 000 & 000 \end{bmatrix}$$

L'un des facteurs d'optimisation les plus importants est le temps de désassemblage, qui est divisé en deux parties, le temps d'exécution du désassemblage (td) et le temps de préparation pour le désassemblage (tprep). Le tprep comprend plusieurs éléments : (i) le temps de changement de direction ; (ii) le temps de changement d'outil de désassemblage qui comprend le temps de manipulation et le temps de transport ; (iii) le temps de manipulation du produit assemblé. Le temps de changement de direction quant à lui est constitué de deux valeurs : 15 secondes pour un changement de direction de 90° et 30 secondes pour un changement de direction de 180°. De même, le temps de changement d'outil comprend deux valeurs : un temps de transport de l'outil estimé à 13 secondes et un temps de traitement de changement d'outil estimé à 10 secondes. L'opération de désassemblage qui nécessite une presse prend environ 27 secondes pour transporter le système vers la presse. Le temps de manipulation manuelle du produit à désassembler est classé en trois catégories : faible (H-L), moyen (H-M) et élevé (H-H). La manipulation manuelle prend 4 secondes pour H-L, 7 secondes pour H-M et 11 secondes pour H-H. Ces paramètres tprep pour illustrer la démarche ont été inspirés de (Aicha et al., 2022b) et synthétisés dans le Tableau 5.

Tableau 5: Le temps opérationnel et de préparation pour le démontage

Opération	*Acronyme*	*Type de préparation*	*Durée du processus (secondes)*
Tourner le mécanisme suivant la direction souhaitée	0° 90° 180°	Direction	0° => 0 90°=>15 180°=>30
Processus de changement d'outil	Tt	Outil	10
Transport de l'outil convenable de l'armoire outil	T	Mouvement	13
Transport de l'ensemble/ sous ensemble à la presse	M	Mouvement	27
Manipulation manuelle	H-L H-M H-H	H-L : Manipulation Faible 4	H-M : Manipulation moyenne 7	H-H : Manipulation Forte 11

Le traitement des collisions donne 22 scénarios de désassemblage possibles représentés dans le

Error! Reference source not found..

En effet, ce tableau illustre la liste des SD planifiées, notées aussi SD, physiquement réalisables, pour préparer les entrées du QN, Ces SD sont obtenues grâce au traitement de la matrice de collision de l'ensemble du mécanisme. Le traitement de la matrice de collision est présenté dans la Figure 15. Le processus automatisé est détaillé dans l'algorithme d'analyse des données CAO présenté dans **Error! Reference source not found.**.

Tableau 6: Liste des séquences de désassemblage (SD) réalisables

PSD1	5(x+)	4(x+)	3(x+)	2(x+)	1(x+)
PSD 2	5(x+)	3(x+)	4(x+)	2(x+)	1(x+)
PSD 3	1(x-)	2(x-)	4(x-)	3(x-)	5(x-)
PSD 4	1(x-)	2(x-)	3(x-)	4(x-)	5(x-)
PSD 5	5(x+)	4(x+)	3(x+)	2(x+)	1(x-)
PSD 6	5(x+)	3(x+)	4(x+)	2(x+)	1(x-)
PSD 7	5(x+)	4(x+)	3(x+)	1(x-)	2(x-)
PSD 8	5(x+)	3(x+)	4(x+)	1(x-)	2(x-)
PSD 9	5(x+)	4(x+)	3(x+)	1(x-)	2(x+)
PSD 10	5(x+)	3(x+)	4(x+)	1(x-)	2(x+)
PSD 11	5(x+)	4(x+)	1(x-)	2(x-)	3(x-)
PSD 12	5(x+)	3(x+)	1(x-)	2(x-)	4(x-)
PSD 13	5(x+)	1(x-)	2(x-)	3(x-)	4(x-)
PSD 14	5(x+)	1(x-)	2(x-)	4(x-)	3(x-)
PSD 15	1(x-)	2(x-)	3(x-)	4(x-)	5(x+)
PSD 16	1(x-)	2(x-)	4(x-)	3(x-)	5(x+)
PSD 17	1(x-)	2(x-)	3(x-)	5(x+)	4(x+)
PSD 18	1(x-)	2(x-)	4(x-)	5(x+)	3(x+)
PSD 19	1(x-)	2(x-)	5(x+)	4(x+)	3(x+)
PSD 20	1(x-)	2(x-)	5(x+)	3(x+)	4(x+)
PSD 21	1(x-)	5(x+)	3(x+)	4(x+)	2(x+)
PSD 22	1(x-)	5(x+)	4(x+)	3(x+)	2(x+)

4.2 GENERATION DE LA MATRICE DE RECOMPENSES BASEE SUR LES DONNEES DE L'ASSEMBLAGE

À partir du Tableau 6, l'algorithme de traitement de la matrice de collision extrait tous les états possibles (Tableau 7) utilisés pour générer la matrice de récompense.

Le premier état représente l'assemblage complet de toutes les pièces, et les cinq derniers états

représentent tous les états finaux possibles pour un désassemblage complet (FV) avec une seule pièce à la fin du processus de désassemblage, ce qui signifie que l'état objectif est l'un des états de S13 jusqu'à S17.

Dans le cas où la SD doit être optimisée pour une maintenance préventive, les pièces d'usure sont les pièces (2) et (5). Une fois ces pièces démontées le processus s'arrête dans l'un des états cibles (S1, S6 et S10).

Tableau 7: Liste des états physiquement réalisables

Etat physiquement possible	Pièce liées
S0	{1,2,3,4,5}
S1	{1,2,3,4}
S2	{2,3,4,5}
S3	{1,2,3}
S4	{1,2,4}
S5	{2,3,4}
S6	{3,4,5}
S7	{1,2}
S8	{2,3}
S9	{2,4}
S10	{3,4}
S11	{3,5}
S12	{4,5}
S13	{1}
S14	{2}
S15	{3}
S16	{4}
S17	{5}

Le nombre d'états de désassemblage physiquement possibles extraits représente la taille de la matrice de récompense [R] : pour cet exemple illustratif, la taille de la matrice [R] est donc [18 x 18]. Une fois l'extraction automatique des états d'assemblage physique terminée, la matrice de récompense sera automatiquement remplie avec les valeurs de récompense définies initialement : (-1) pour une action impossible, (0) pour une action possible sans avantages et (100) pour la meilleure action avec avantages.

[R]=

		S0	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	S16	S17
{1,2,3,4,5}	S0	0	100	100	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
{1,2,3,4}	S1	-1	0	-1	100	100	100	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
{2,3,4,5}	S2	-1	-1	0	-1	-1	100	100	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
{1,2,3}	S3	-1	-1	-1	0	-1	-1	-1	100	100	-1	-1	-1	-1	-1	-1	-1	-1	-1
{1,2,4}	S4	-1	-1	-1	-1	0	-1	-1	100	-1	100	-1	-1	-1	-1	-1	-1	-1	-1
{2,3,4}	S5	-1	-1	-1	-1	-1	0	-1	-1	100	-1	100	-1	-1	-1	-1	-1	-1	-1
{3,4,5}	S6	-1	-1	-1	-1	-1	-1	0	-1	-1	-1	100	100	100	-1	-1	-1	-1	-1
{1,2}	S7	-1	-1	-1	-1	-1	-1	-1	0	-1	-1	-1	-1	-1	100	100	-1	-1	-1
{2,3}	S8	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	-1	-1	-1	-1	100	100	-1	-1
{2,4}	S9	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	-1	-1	-1	100	-1	100	-1
{3,4}	S10	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	-1	-1	-1	100	100	-1
{3,5}	S11	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	-1	-1	100	-1	100
{4,5}	S12	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	-1	-1	100	100
{1}	S13	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	100	-1	-1	-1	-1
{2}	S14	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	100	-1	-1	-1
{3}	S15	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	100	-1	-1
{4}	S16	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	100	-1
{5}	S17	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	100

Figure 21: La matrice de récompense [R] basée sur les états physiquement réalisables

5 EXECUTION DU QN : RESULTATS ET INTERPRETATION

La matrice [R'] guide l'algorithme d'apprentissage par renforcement pour générer l'un des 22 épisodes de démontage (Tableau 8), qui sont classés en fonction de la valeur du score donné par la fonction fitness.

Tableau 8: Liste des épisodes physiquement réalisables

Episode de démontage	Etat initial	2ème état	3ème	4ème état	Etat final	FFV	FMP
E1	S0	S1	S3	S7	S13	0.6429	0.5944
E2	S0	S1	S3	S7	S14	0.6417	0.5934
E3	S0	S1	S3	S8	S14	0.5941	0.5553
E4	S0	S1	S3	S8	S15	0.5901	0.5521
E5	S0	S1	S4	S7	S13	0.6613	0.6091
E6	S0	S1	S4	S7	S14	0.5993	0.5595
E7	S0	S1	S4	S9	S14	0.6085	0.5668
E8	S0	S1	S4	S9	S16	0.5981	0.5585
E9	S0	S1	S5	S8	S14	0.5573	0.5259
E10	S0	S1	S5	S8	S15	0.4953	0.4763
E11	S0	S1	S5	S10	S15	0.5945	0.5556
E12	S0	S1	S5	S10	S16	0.5885	0.5508
E13	S0	S2	S5	S8	S14	0.5977	0.5582
E14	S0	S2	S5	S8	S15	0.5157	0.4926
E15	S0	S2	S6	S10	S15	0.6477	0.5982
E16	S0	S2	S5	S10	S16	0.5965	0.5572
E17	S0	S2	S5	S10	S15	0.5965	0.5572
E18	S0	S2	S5	S10	S16	0.5965	0.5572
E19	S0	S2	S6	S11	S15	0.5873	0.5499
E20	S0	S2	S6	S11	S17	0.6597	0.6078
E21	S0	S2	S6	S12	S15	0.5993	0.5595
E22	S0	S2	S6	S12	S17	0.6493	0.5995

Dans le Tableau 8, les poids ont été choisis égaux dans la fonction fitness (FF), de sorte que le processus d'optimisation soit identique vis-à-vis de tous les paramètres intégrés dans l'équation 1, c'est-à-dire que $\alpha=\beta=\delta=\gamma$ dans le cas de démontage de FV et $\alpha=\beta=\delta=\gamma=\lambda$ dans le cas de démontage pour la MP. Cela signifie que chaque paramètre contribue de manière équilibrée à la valeur globale de la fonction d'optimisation.

Cela peut être représenté graphiquement par la Figure 22. Dans cette figure, les valeurs de la fonction d'optimisation sont affichées pour chaque SD à la fois pour la maintenance préventive (FMP) et pour le démontage en fin de vie (FFV). La fitness d'un épisode est égal à la somme des fitness de tous les états divisés par le nombre d'états effectué pour atteindre l'état cible.

La valeur moyenne de la fonction fitness pour l'épisode de démontage en fin de vie ($\lambda=0$) est (FFV

moy) et la valeur moyenne de la fonction fitness pour le démontage pour la maintenance préventive (MP) est (FMP moy). L'épisode le plus optimal est alors celui ayant la valeur de la fonction fitness la plus élevée. A noter qu'il est possible de trouver plusieurs SD optimales et de modifier le classement des SD optimales en modifiant les valeurs des poids dans la fonction fitness, par exemple, en augmentant la valeur d'un des poids pour donner la priorité au paramètre associé.

Équation 4 :

$$F\ moy = \frac{\sum_0^i F(Si)}{m}$$

Où :

F moy : est le fitness moyen de l'épisode (scénario) de démontage.

m : est le nombre des états tout au long de l'épisode (de l'état initial jusqu'à l'état cible).

F(Si) : est la valeur de fitness de chaque état

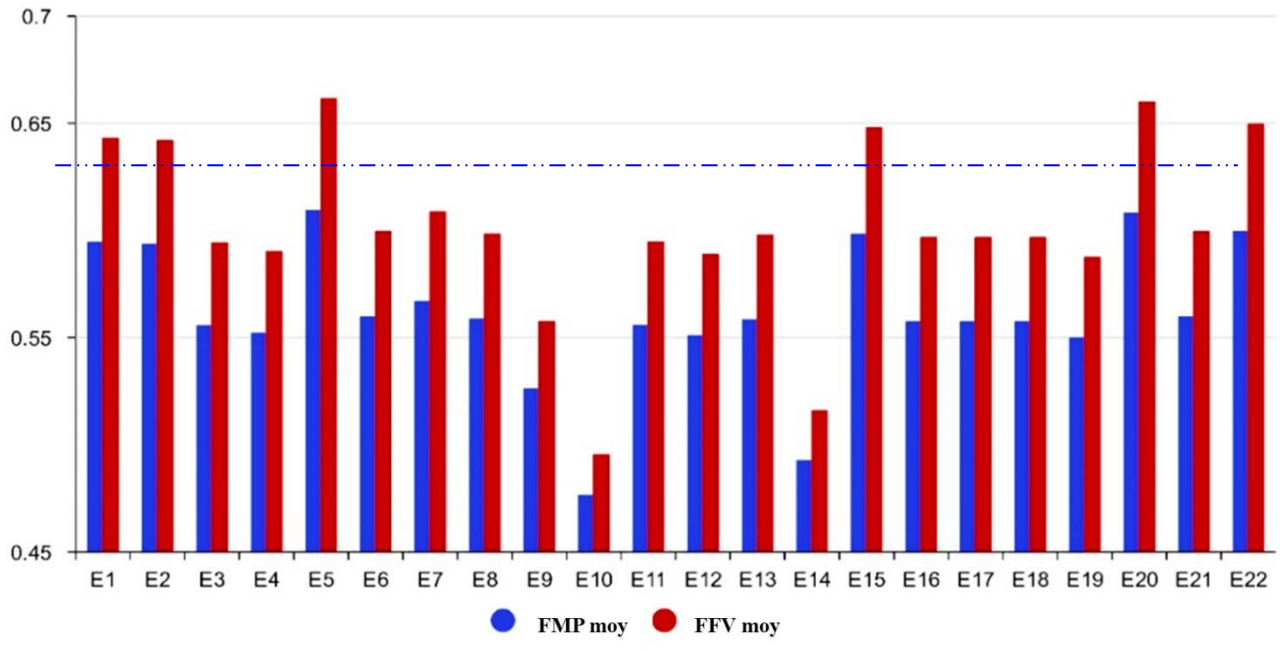


Figure 22: Évaluation des épisodes de désassemblage basée sur des valeurs de pondération égales pour l'optimisation de tous les paramètres introduits.

Dans le cas de la maintenance préventive (FFPM), les poids utilisés sont α=β=δ=γ=λ=0.2 Ça signifie que chaque paramètre contribue à 20% à la valeur globale de la fonction d'optimisation.

$$F(etat, action) = \alpha \times \left(1 - \left(\frac{V_{(i)}}{Vmax}\right)\right) + \beta \times \left(1 - \left(\frac{t_{(i)}}{t_{max}}\right)\right) + \gamma \times D(i) + \delta \times T(i) + \lambda P(i)$$

Dans le cas du démontage en fin de vie (FFV), les poids utilisés sont α=β=δ=γ=0.25 et λ=0 (facteur lié à la maintenance préventive), ce qui signifie que chaque paramètre contribue à 25% à la valeur globale de la fonction d'optimisation. Comme la Figure 22 représente les valeurs de la fonction d'optimisation pour chaque SD, ce qui permet de comparer les différentes séquences et de déterminer celle qui a la valeur de la fonction d'optimisation la plus élevée.

Pour le processus de désassemblage en fin de vie (FV), les épisodes E1, E2, E5, E15, E20 et E22 sont les meilleurs épisodes qui optimisent également tous les paramètres de désassemblage. Pour le processus de maintenance préventive (MP), les épisodes E1, E2, E5, E15, E20 et E22 sont les meilleurs épisodes. Dans cet exemple, l'optimisation égale des paramètres donne les mêmes meilleurs épisodes pour les processus de désassemblage (FV) et (MP). Pour plus de détails, le processus d'évaluation pour l'accès aux pièces d'usure et aux plus petites pièces en priorité est présenté dans le Tableau 9.

Tableau 9: Les meilleurs épisodes de démontage pour les pièces d'usure et les priorités des pièces les plus petites.

Episodes	*Scénario de démontage*	*Changement de direction*	*Changement d'outil*	*Accès aux pièces d'usure*	*Accès au petites pièces*
E1	5(x+) 4(x+) 3(x+) 2(x+) 1(x+)	0	2	++	++
E2	5(x+) 4(x+) 3(x+) 1(x-) 2(x+)	1	3	++	++
E5	5(x+) 4(x+) 3(x+) 2(x+) 1(x-)	0	2	++	++
E15	1(x-) 2(x-) 5(x+) 4(x+) 3(x+)	1	2	++	+
E20	1(x-) 2(x-) 4(x-) 3(x-) 5(x-)	0	2	+	+
E22	1(x-) 2(x-) 3(x-) 4(x-) 5(x-)	0	2	+	+

Où :

(++) : Pour l'épisode qui commence par une pièce d'usure ou une plus petite pièce.

(+) : Pour l'épisode qui donne la priorité aux plus petites pièces et aux pièces d'usure, mais ne commence pas ces dernières.

La procédure d'optimisation peut être exprimée par la priorité des paramètres à optimiser. Par exemple, pour la maintenance préventive, il est recommandé de donner la priorité à l'accès aux pièces d'usure. Ce processus est réalisé en augmentant la valeur de pondération des pièces d'usure ($\delta=\beta=\alpha=\gamma=0{,}15$, $\lambda=0{,}4$). Ce processus est le même pour les autres paramètres. La Figure 23 représente les meilleurs scénarios (épisodes) de désassemblage (E) pour chaque priorité de paramètre.

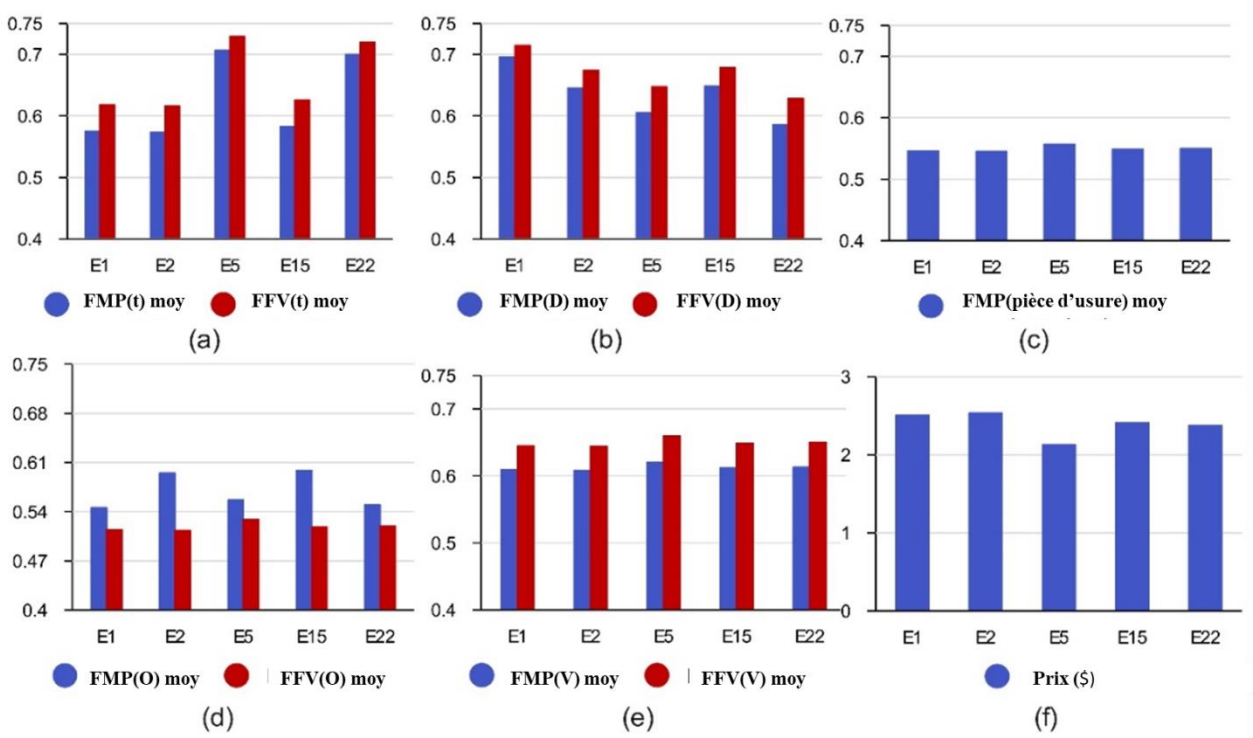


Figure 23: Meilleurs scénarios de désassemblage pour l'optimisation chaque paramètre tout seul

Dans cette figure on trouve :

- La Figure 23 (a) illustre l'évolution des meilleurs épisodes (E1, E2, E5, E15 et E22) respectivement en considérant la maintenance préventive (FMP(t) moy) et la fin de vie (FFV(t) moy) en favorisant la minimisation du temps de désassemblage comme objectif principal.
- La Figure 23 (b) représente l'évolution des meilleurs épisodes en considérant la maintenance préventive (FMP(D) moy) et la fin de vie (FFV(D) moy) en favorisant la minimisation du changement de direction de désassemblage comme objectif principal.
- La Figure 23 (c) illustre l'évolution des meilleurs épisodes en considérant la maintenance préventive (FMP (pièce d'usure) moy) en favorisant l'accès aux pièces d'usure comme objectif principal.
- La Figure 23 (d) décrit l'évolution des meilleurs épisodes en considérant la maintenance préventive (FMP(O) moy) et la fin de vie (FFV(O) moy) en favorisant la minimisation du changement d'outil de désassemblage comme objectif principal.

- La Figure 23 (c) représente l'évolution des meilleurs épisodes en considérant la maintenance préventive (FMP(V) moy) et la fin de vie (FFV(V) moy) en favorisant le désassemblage des plus petites pièces en priorité comme objectif principal.
- La Figure 23 (f) représente les coûts des meilleurs épisodes pour un coût de main-d'œuvre estimé à 30 $/heure.

Parmi les cinq meilleurs épisodes réalisables, l'épisode le plus optimal peut changer en fonction du paramètre à optimiser en priorité.

La Figure 24 décrit le temps et le coût des épisodes de désassemblage. L'épisode de désassemblage optimal est celui qui prend le moins de temps et coûte le moins cher. Ainsi, l'épisode E5 est le plus optimal, en deuxième lieu vient l'épisode E20.

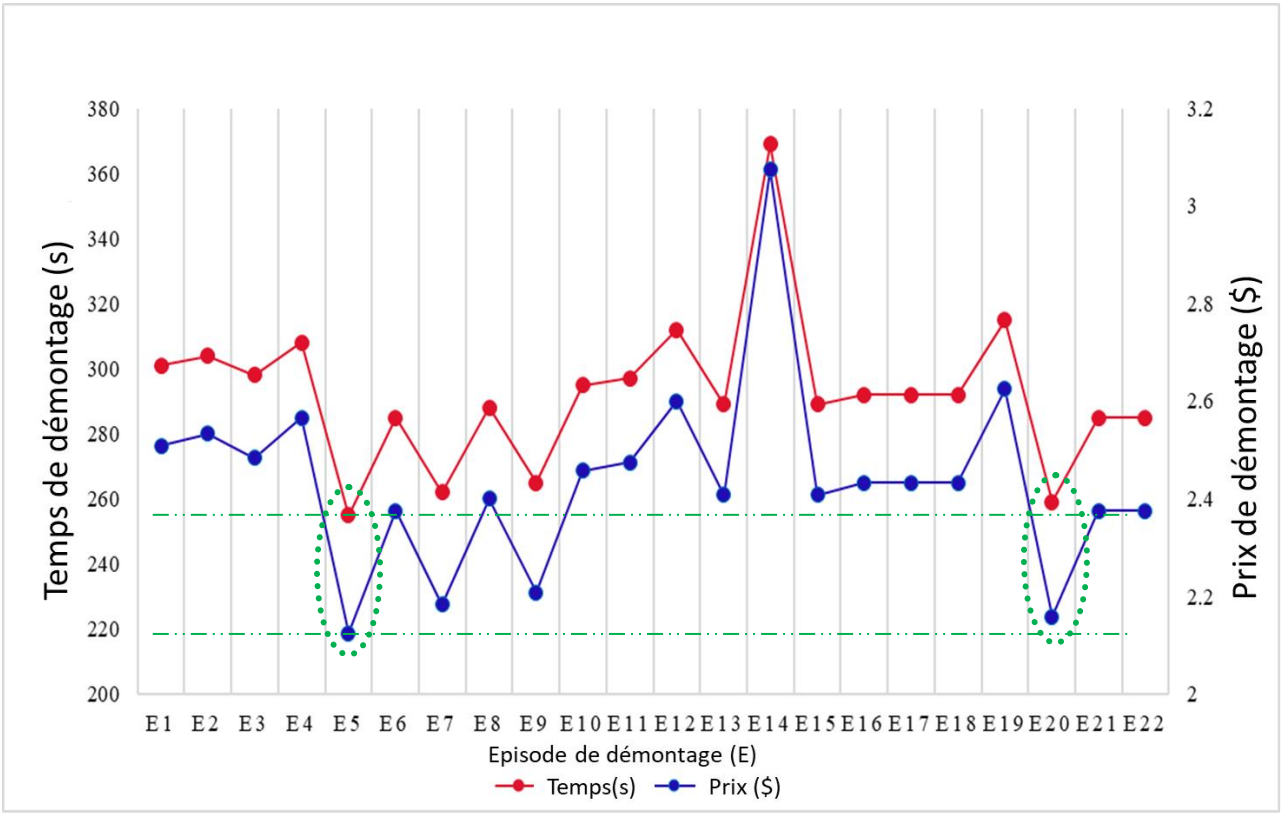


Figure 24: Comparaison du temps et du coût des SD par épisode

Tableau 10: Meilleur épisode de désassemblage (E5)

Scénario de désassemblage E5					
Etats	S0	S1	S2	S3	S4
Composants de l'ensemble	(1,2,3,4,5)	(1,2,3,4)	(1,2,3)	(1,2)	(1)
E5	5	4	3	2	1

7 CONCLUSION

Dans ce chapitre, nous avons exposé la première contribution de cette thèse, qui consiste en l'optimisation des SD à l'aide d'un algorithme RL. Basée sur les données géométriques de l'assemblage, l'approche proposée cherche à optimiser la SD en utilisant l'algorithme Q-Network. La méthode d'optimisation proposée est appliquée à deux stratégies de désassemblage : (i) planification des SD pour la maintenance préventive (PSDMP) et (ii) planification des SD pour le désassemblage en fin de vie (PSDFV). Le processus d'optimisation intègre cinq paramètres : (1) minimiser les changements d'outil de désassemblage, (2) minimiser les changements de direction de désassemblage, (3) optimiser le temps de désassemblage incluant le temps de préparation et de traitement, ainsi que deux paramètres qui relèvent des pratiques industrielles : (4) donner la priorité au désassemblage des plus petites pièces et (5) faciliter l'accès aux pièces d'usure. L'algorithme RL développé a été implémenté dans l'environnement Python en utilisant des bibliothèques d'intelligence artificielle.

Pour ce faire, nous avons d'abord présenté la méthode en détaillant les étapes et en exposant les hypothèses prises en compte lors de la mise en œuvre de cette approche. Pour une meilleure compréhension de l'approche proposée, nous avons illustré celle-ci à l'aide d'un exemple concret contenant cinq pièces. Par la suite, nous avons analysé les résultats obtenus par l'application de l'approche sur cet exemple, afin de démontrer sa faisabilité.

Le chapitre suivant portera sur la mise en œuvre informatique et la validation de ces travaux par comparaison avec d'autres approches reposant sur des algorithmes d'intelligence artificielle, tels que l'algorithme génétique et l'optimisation par colonie de fourmis.

CHAPITRE 3 : MISE EN ŒUVRE INFORMATIQUE ET VALIDATION

1 INTRODUCTION

Ce chapitre présente l'implémentation informatique et la validation de l'approche proposée. La première partie porte sur l'environnement et les outils utilisés, ainsi que sur le processus d'extraction des données du modèle CAO utilisé. La deuxième partie présente une comparaison de l'approche proposée avec deux autres approches appliquées sur le même exemple démonstratif. Après avoir présenté les deux méthodes issues de la littérature, leurs résultats et détails de l'implémentation informatique seront exposés. Puis, ces résultats seront comparés avec les résultats de l'approche proposée. La dernière partie de ce chapitre, expose une synthèse sur les principaux avantages de l'approche proposée ainsi que les principales limitations.

2 IMPLEMENTATION INFORMATIQUE

2.1 ENVIRONNEMENT INFORMATIQUE UTILISE

Pour mettre en œuvre le fruit de ce travail, deux environnements de développement informatique ont été utilisés : les API (Application Programming Interface) de SolidWorks© et Python© (Figure 25). Dans ce qui suit, chaque environnement sera présenté.

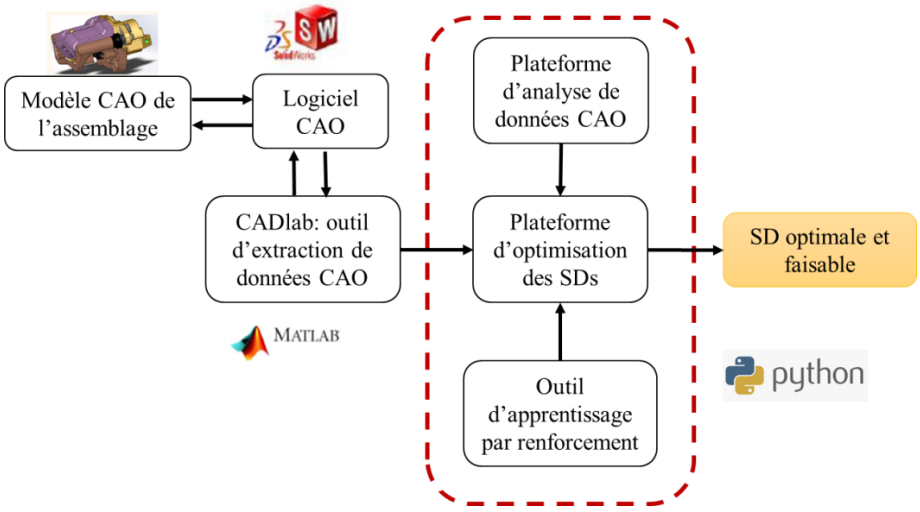


Figure 25: Interopérabilité des logiciels utilisés pour l'implémentation de l'outil d'optimisation des SD

L'extraction des données a été implémentée dans le logiciel de CAO SolidWorks©, grâce à ses API. Le choix de cet outil de modélisation est justifié par l'accessibilité à ses fonctionnalités classiques et particulièrement à ses API. En exploitant les API de SolidWorks, il est possible de récupérer toutes les données relatives au modèle CAO nécessaires aux approches développées.

Le langage de programmation orienté objet Python est un outil polyvalent largement utilisé dans le domaine de l'analyse de données et de l'apprentissage par renforcement. Python a été utilisé dans ce travail par le biais de l'interface Spyder Python, également connue sous le nom de Spyder IDE (Integrated Development Environment). Il s'agit d'un environnement de développement intégré conçu principalement pour la science des données, l'analyse numérique et les tâches de programmation Python. Cette interface est particulièrement adaptée aux scientifiques et aux « data scientists », grâce à ses fonctionnalités de pointe pour l'analyse et la science des données.

Python a été utilisé pour l'implémentation des travaux de cette thèse dans les contextes suivants :

- Analyse des données :
    - NumPy : utilisé pour effectuer des opérations numériques sur des tableaux multidimensionnels. Il est essentiel pour la manipulation et le traitement efficace des données.
    - Matplotlib : utilisé pour créer des visualisations graphiques, comme des graphiques et des tracés, afin de mieux comprendre les données.
    - Pandas : utile pour importer, nettoyer, transformer et analyser des données à l'aide des objets DataFrame.
    - Random : fournit des fonctions pour générer des nombres aléatoires, ce qui peut être utile pour la génération de données synthétiques ou pour des expériences (May et al., 2022).
- Apprentissage par renforcement (Q-learning) :
    - Q-learning : cet algorithme d'apprentissage par renforcement ,est un script développé durant notre approche basée sur une équation de qualité Q, permet à un agent d'apprendre une politique optimale en explorant un environnement. L'agent apprend une fonction Q, qui estime la valeur d'action d'état (Q-value), pour prendre des décisions basées sur les récompenses futures attendues.
    - NumPy : utilisé pour gérer les calculs numériques nécessaires à l'implémentation de l'algorithme Q-learning, notamment pour la mise à jour des valeurs Q.
    - Random : utilisé pour générer des actions aléatoires lors de l'exploration de l'environnement.
    - Matplotlib : sert à visualiser l'apprentissage de l'agent, en traçant l'évolution des valeurs Q, des récompenses accumulées, etc (Kwiatkowski et al., 2022).
- Script d'extraction de la séquence de démontage optimale et faisable.

2.2 PRESENTATION DE L'EXEMPLE DE VALIDATION

L'exemple choisi pour valider l'approche proposée est un étrier de frein. L'étrier de frein joue un

rôle crucial au sein du système de freinage d'un véhicule, agissant comme une force pressante sur les plaquettes de frein en contact avec le disque. Ce mécanisme génère le frottement essentiel pour ralentir ou immobiliser la rotation de la roue. Il existe une variété d'étriers de frein, incluant les modèles fixes et flottants, qui diffèrent en termes de qualité et de performance, influençant par conséquent leur coût.

L'étrier de frein est composé de 16 pièces représentées dans le **Error! Reference source not found.**.

Tableau 11: Nomenclature des composants du modèle CAO (Kheder et al., 2015b)

Composants	Nombre	Nomenclature	Outil
1	1	Étrier de frein arrière	G1: Main
2	1	Étrier	G1
3	1	Étrier de frein avant	G1
4,5	2	Plaquette de frein	G1
6	1	Piston	G5: Kit repousse piston
7,8	2	Vis CHC	G3: Clé six pans
9,10	2	Vis H	G2: Clé fourche 1
11	1	Colonnette A	G4: Clé fourche 2
12	1	Colonnette B	G4
13,14	2	Joint de colonnette	G1
15	1	Joint	G6: Tournevis plat
16	1	Joint carré	G1

Dans cet ensemble, étant donné qu'il y a deux pièces d'usure avec une fréquence d'entretien élevée (patins de disque (4,5)), ce paramètre devrait être considéré comme un paramètre principal pour optimiser le processus de désassemblage.

La Figure 26 illustre les différents composants avec leur positionnement suivant les directions x,y,z suivant lesquelles les matrices d'interférence seront générées respectivement [Ix], [Iy], [Iz].

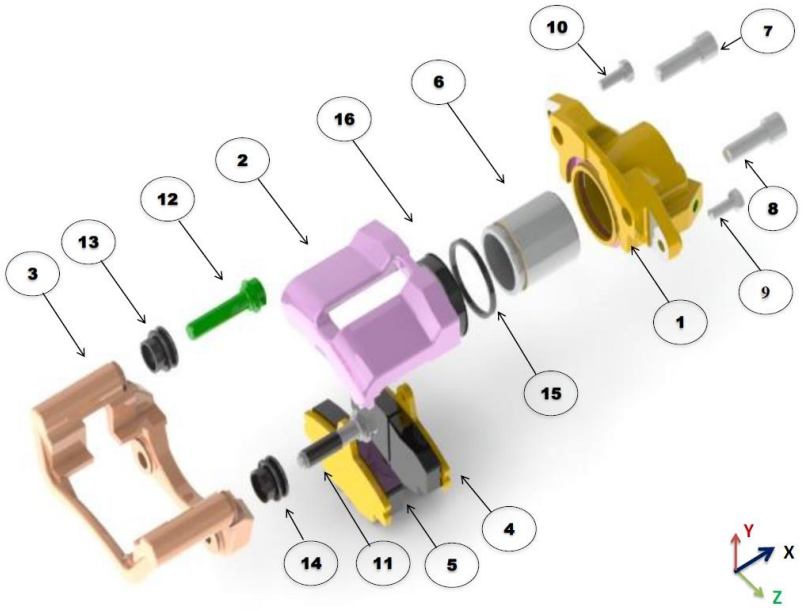


Figure 26 : Modèle CAO de l'étrier de frein (Kheder et al., 2015b)

Comme pour l'approche proposée dans cette thèse, Khedher et al. définissent la matrice de collision comme entrée principale dans leurs approches d'optimisation utilisant les techniques des algorithmes génétiques (GA : Genetic Algorithm) et de colonies de fourmis (ACO : Ant Colony Optimization). Les deux approches sont discutées dans la section 3.

Au préalable, nous introduisons dans la section suivante la démarche d'extraction des données CAO, nécessaire pour notre approche de génération des SD optimales.

2.3 EXTRACTION ET IMPORTATION DES DONNEES CAO

Avant de commencer la phase de génération et d'optimisation des SD, il est essentiel de lancer la phase préliminaire d'extraction des données à partir du modèle CAO. Cette étape s'effectue en choisissant le modèle d'assemblage via le menu contextuel accessible depuis l'interface principale, comme illustré dans la Figure 27. L'exemple d'étrier de frein est déjà disponible dans ce menu, car c'est l'un des mécanismes les plus récents pris en charge par cet outil. Si le mécanisme que nous souhaitons traiter ne figure pas dans le menu contextuel, il nous faudra parcourir tous les assemblages disponibles et sélectionner le mécanisme en question en utilisant le bouton "Parcourir le modèle d'assemblage". Une fois que le modèle d'assemblage est sélectionné, il suffit de cliquer sur le bouton "Extraire les données" dans la même section.

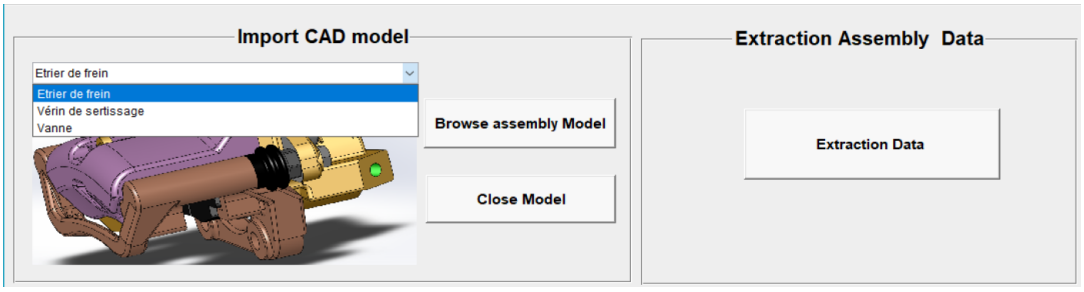


Figure 27: Interface d'extraction des données CAO du modèle CAO (Kheder et al., 2015b)

Une nouvelle fenêtre, comme celle présentée dans la Figure 28, s'affiche, permettant de procéder à l'extraction des données topologiques et géométriques. Nous disposons ainsi d'une base de données prête à être utilisée lors de la phase d'optimisation. Cette base de données est extraite sous forme de fichiers MATLAB et Excel.

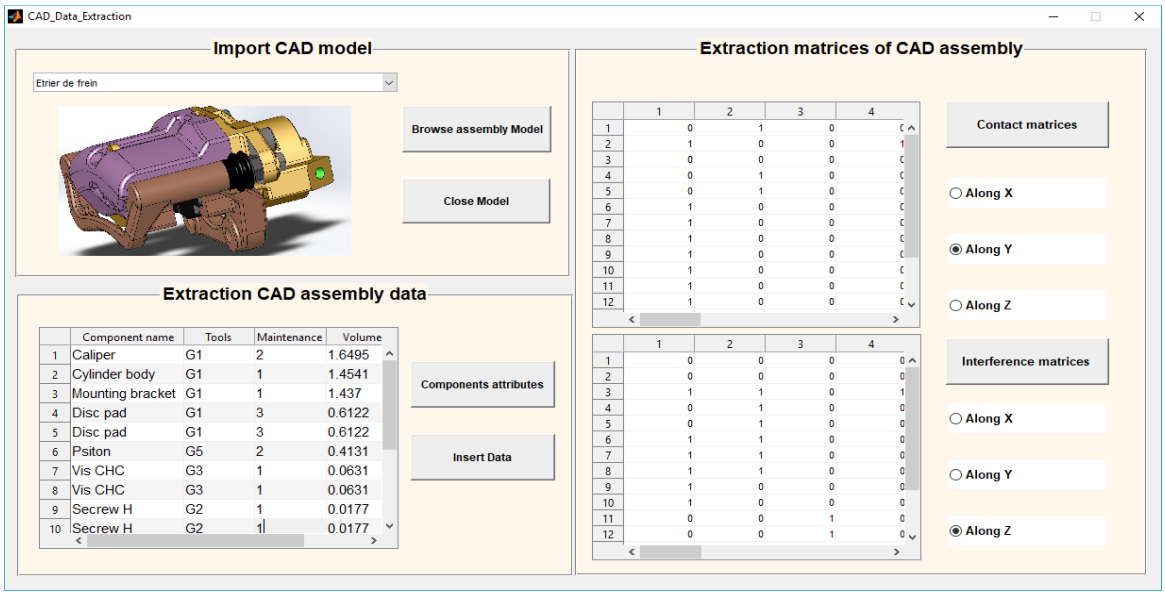


Figure 28: Interface d'extraction des données topologiques et géométriques du modèle CAO (Kheder et al., 2015b)

Les fichiers Excel extraits seront utilisés dans l'interface de programmation Spyder python pour la préparation de la matrice de récompense comme détaillé dans le chapitre 2. La section suivante présentera la comparaison des résultats trouvés par ce travail à deux autres travaux de la littérature.

3 COMPARAISON ET VALIDATION

Cette section compare l'outil d'optimisation des SD pour la fin de vie et la maintenance en utilisant l'apprentissage par renforcement avec les deux approches d'optimisation des SD utilisant respectivement le GA et le ACO. Ces approches sont validées sur le même exemple démonstratif présenté précédemment.

3.1 OPTIMISATION DES SD AVEC LE GA

Un GA est un type d'algorithme d'intelligence artificielle qui s'inspire du processus de sélection naturelle pour résoudre des problèmes d'optimisation ou de recherche. Il utilise des techniques de reproduction, de mutation et de sélection pour évoluer et améliorer des solutions potentielles au fil du temps.

Après l'extraction des données du modèle CAO, l'algorithme démarre en créant une population initiale P de solutions, chaque solution étant représentée par un chromosome. Dans ce cas, ces chromosomes sont structurés en quatre sections : La première section détaille la séquence des opérations de désassemblage. La deuxième section spécifie la direction dans laquelle chaque composant doit être démonté, avec des coordonnées telles que {x, y, z, -x, -y, -z}. La troisième section indique les outils utilisés pour le désassemblage de chaque composant. La quatrième section informe sur la maintenabilité des composants (préventive ou corrective), tandis que la cinquième section décrit le volume occupé par chaque composant. Dans l'approche basée sur le GA (Figure 29), chaque chromosome de la population est divisé en cinq parties de longueurs égales pour représenter ces informations.

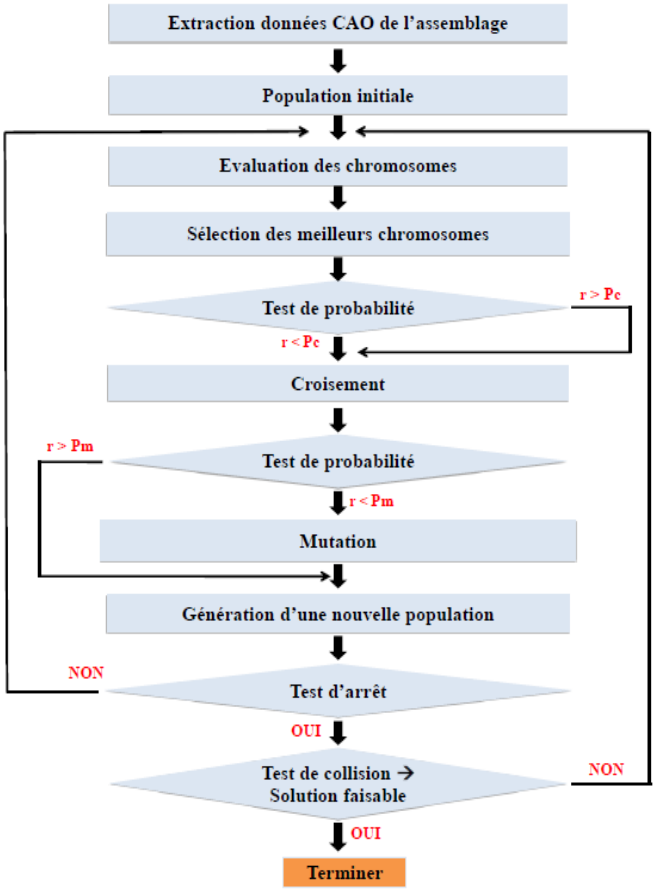


Figure 29: Organigramme du GA proposé (Kheder et al., 2015b)

Pour l'exemple démonstratif de l'étrier de frein les caractéristiques des composants sont présentées dans le Tableau 12.

Tableau 12: Caractéristiques des composants de l'étrier de frein

Composant	Maintenabilité	Direction	Outil	Volume(mm^3) 10^5
1	2	X	G1	1.6495
2	1	Y	G1	1.4541
3	1	-X	G1	1.437
4	3	Y	G1	0.6122
5	3	Y	G1	0.6122
6	2	-X	G5	0.4131
7	1	X	G3	0.0631
8	1	X	G3	0.0631
9	1	X	G2	0.0177
10	1	X	G2	0.0177
11	1	X	G4	0.0769
12	1	X	G4	0.0729
13	1	X	G1	0.0089
14	1	X	G1	0.0089
15	2	-X	G6	0.0015
16	2	-X	G1	0.0019

La représentation d'un chromosome est donnée par le Tableau 13.

Tableau 13: Exemple d'un chromosome

Séquence	7	8	2	4	5	9	10	1	11	14	12	13	3	16	6	15
Direction	X	X	Y	Y	Y	X	X	X	X	X	X	X	-X	-X	-X	-X
Outil	G3	G3	G1	G1	G1	G2	G2	G1	G4	G1	G4	G1	G1	G1	G5	G6
Maintenabilité	1	1	1	3	3	1	1	2	1	1	1	1	1	2	2	2
Volume (mm3)	06.31	06.31	145.41	61.22	61.22	1.77	1.77	164.95	7.69	0.89	0.0729	0.89	143.7	0.19	41.31	0.15

Comme illustré dans l'exemple du chromosome précédent, le composant 4, désigné sous le nom de "plaquette de frein", sera l'objet d'une maintenance préventive. Il doit être démonté après le composant 2 et avant le composant 3. Le démontage s'effectue dans la direction Y à l'aide de l'outil G1 et il occupe un volume de 61,22 mm3. Le Tableau 14 présente la population initiale correspondante. La taille de cette population est notée "n", et pour cet exemple, nous choisissons n=14.

Tableau 14: Population initiale générée par l'AG

Population Initiale													
S1	***S2***	***S3***	***S4***	***S5***	***S6***	***S7***	***S8***	***S9***	***S10***	***S11***	***S12***	***S13***	***S14***
7	8	7	7	10	9	9	9	10	7	7	10	8	7
8	7	8	8	9	8	7	10	9	8	8	9	7	8
2	9	2	2	7	10	10	7	7	2	2	7	9	2
4	10	9	4	8	7	8	8	8	4	4	8	10	4
5	1	10	5	1	2	1	2	2	5	5	2	1	5
9	16	1	9	2	1	2	4	3	9	10	4	2	10
10	6	4	10	11	16	4	5	4	10	9	5	5	9
1	15	16	1	12	6	5	1	5	1	1	1	4	1
11	2	11	16	13	15	16	16	11	16	12	11	13	12
14	4	14	6	14	11	6	11	12	6	13	12	12	13
12	5	12	15	4	14	15	12	13	15	11	14	11	11
13	11	13	11	5	12	11	13	14	11	14	13	14	14
3	14	5	14	3	13	12	14	1	12	3	16	3	16
16	12	3	12	16	3	13	6	16	13	16	3	16	6
6	13	6	13	6	4	14	15	6	14	6	6	6	15
15	3	15	3	15	5	3	3	15	3	15	15	15	3

Pour évaluer la qualité et les performances de leur solution, Khedher et al. ont développé une Fonction Objective (FO) en intégrant des paramètres qui ont une influence directe ou indirecte sur le processus de démontage. Parmi les paramètres susceptibles d'influencer le processus de démontage, on peut citer :

- La performance du démontage au sein de la chaîne de production et son impact sur le processus de recyclage.
- L'impact du type de processus utilisé :
    - Le nombre de directions impliquées dans le processus de démontage.
    - Les éventuelles réorientations pendant le démontage.
    - Les outils spécifiques utilisés lors de l'opération.
    - La facilité de maintenance du composant au sein d'un mécanisme donné.

La fonction objective (FO) est définie comme suit :

Équation 5

$$\boldsymbol{OF = N + \alpha \times M + \beta \times D + \gamma \times T + \mu \times V}$$

Où :

- $\boldsymbol{N}$ est le nombre de pièces du mécanisme.
- $\boldsymbol{M}$ est le facteur de maintenance relatif pour chaque composant, donné par l'équation :

Équation 6

$$M = \sum_{K=1}^{N} \frac{mi}{\sum mi}(N - k + 1)$$

Où

$\boldsymbol{mi}$ prend :

- 1 si aucune maintenance n'est nécessaire pour le composant i.
- 2 : si une maintenance corrective du composant i est nécessaire.
- 3 : si une maintenance préventive du composant i est nécessaire.

$\boldsymbol{D}$: est le facteur de changement de direction :

- 0 : s'il n'y a pas de changement entre deux parties consécutives.
- 1 : s'il y a un changement de 90° entre deux parties consécutives.
- 2 : s'il y a un changement de 180° entre deux pièces consécutives.

$\boldsymbol{T}$: est le facteur de changement d'outil :

- 0 : s'il n'y a pas de changement d'outils entre deux pièces successives.
- 1 : si un changement d'outil est nécessaire entre deux pièces successives.

$\boldsymbol{V}$: est le volume relatif de chaque composant du mécanisme, donné par l'équation 7 :

- Vi : le volume de la pièce étudiée.
- N : le nombre de pièces dans l'assemblage.

Équation 7

$$V = \sum_{i=1}^{N} \frac{Vi}{\sum Vi}(N - i + 1)$$

α, β, γ et μ représentent des coefficients de pondération, qui peuvent être choisis en fonction des

objectifs du planificateur où : |α|+|β|+|ɣ|+|μ| = 1.

Dans la validation de l'approche GA, Khedher et al. ont défini les coefficients de poids comme suit : α = -0,4 et β = ɣ = μ = -0,2. L'évolution de la fonction objective en fonction des SD générés est donnée par la Figure 30.

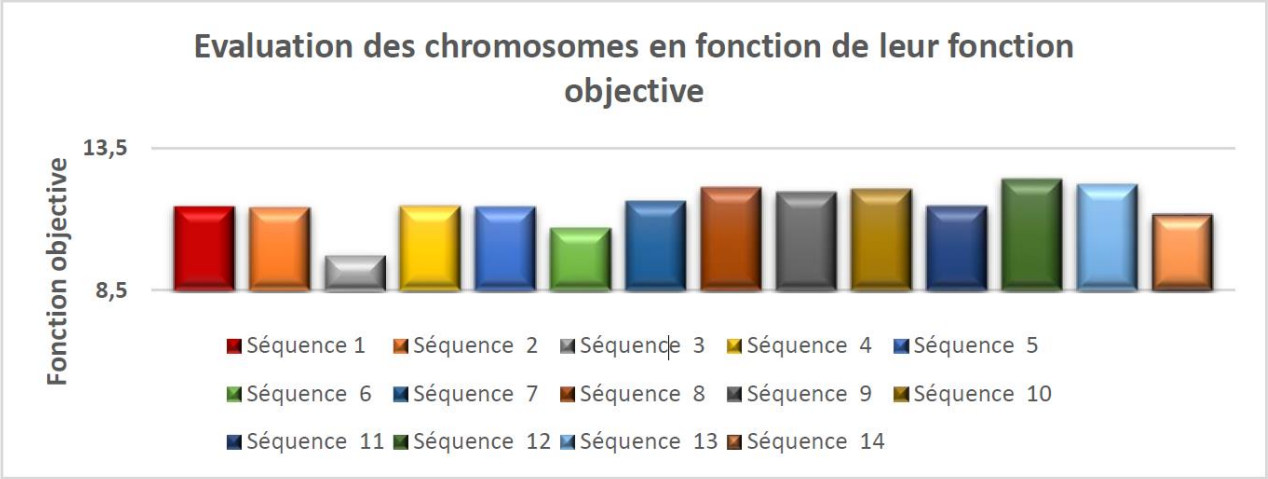


Figure 30: Evaluation des chromosomes (SD) en fonction de leur fonction objective

Pour plus de précision l'algorithme est exécuté en faisant varier le nombre de génération (NG). Le temps d'exécution peut varier entre 0.3 et 46 secondes pour un nombre de génération qui varie respectivement entre 5 et 1000. La Figure 31 montre la variation de temps d'exécution CPU (s) en fonction du nombre de génération (NG).

Nombre de générations	CPU (s)
5	0.324
10	0.355
20	0.426
50	0.677
100	1.251
200	3.058
300	5.695
500	12,92
1000	46,07

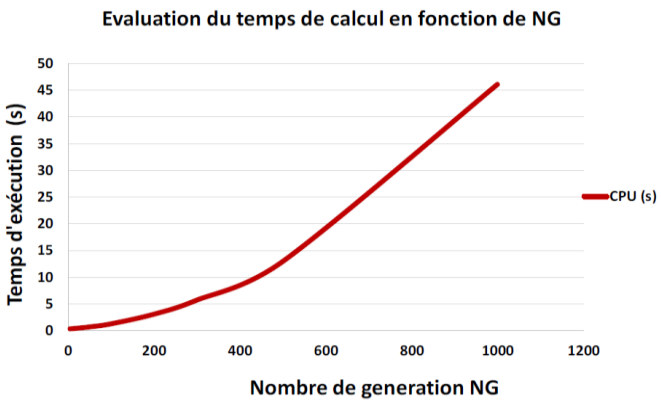


Figure 31: Évolution du temps CPU (s) en fonction du nombre de générations (NG)

Les critères significatifs dans l'exécution du GA sont : La probabilité de croisement (Pc), La probabilité de mutation (Pm) et le nombre R. C'est un nombre aléatoire entre 0 et 1. Il sert à déterminer si un individu subira une mutation ou sera choisi pour le croisement. Si R<Pm une mutation est appliquée à l'individu, et si R<Pc, l'individu est sélectionné pour se croiser. Cette

utilisation de nombres aléatoires est cruciale pour introduire de l'aléatoire dans l'algorithme, prévenant ainsi la convergence vers des solutions optimales. Après plusieurs exécutions et tests de stabilité détaillés dans le travail de Khedher et al., la meilleure solution retenue est caractérisée par **Pc = 0.9**, **Pm = 0.1**. Le temps de calcul CPU est de 1,305s. La SD retenue s'avère faisable et elle est détaillée dans la Figure 32. Le démontage des pièces (4) et (5) qui ont une fréquence de maintenance élevée aura lieu en 6ème et 7ème position dans la SD en jaune.

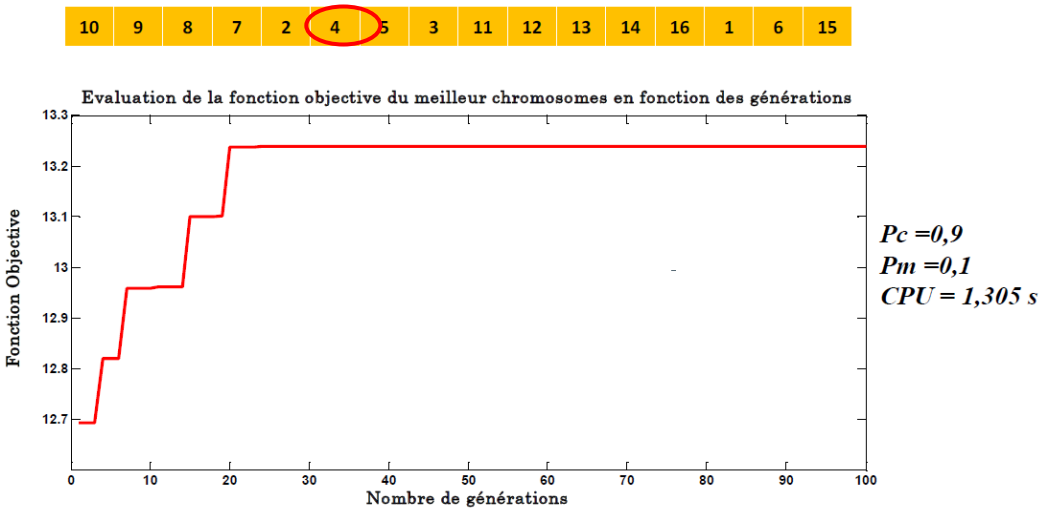


Figure 32: Evolution de la fonction objective en fonction du nombre de génération

Dans un temps d'exécution de 1.305 (s), l'AG peut fournir sa solution optimale, montrant la faisabilité et le niveau d'efficacité de l'approche proposée. Dans la section suivante on discute de la deuxième approche proposée par Khedher et al. pour l'optimisation des SD en utilisant l'ACO.

3.2 OPTIMISATION DES SD AVEC L'OPTIMISATION PAR COLONIE DE FOURMIS

Comme détaillé dans le premier chapitre l'ACO est un algorithme itératif qui repose sur la collaboration de tous les individus qui partagent une connaissance commune. Son but est de diriger d'autres agents vers le chemin le plus court en utilisant une substance chimique appelée phéromone. Khedher et al. ont adapté cette méthode pour l'optimisation des SD. Pour ce faire, la Figure 33 présente les étapes principales de la méthode.

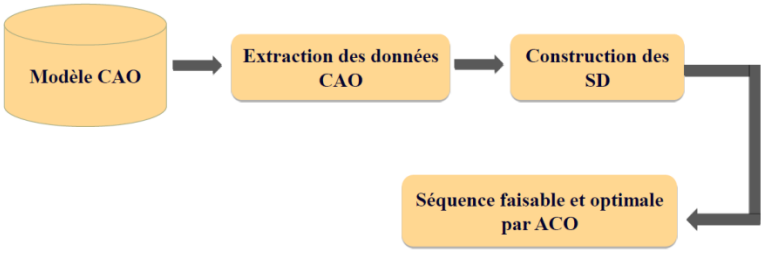


Figure 33: Optimisation des SDS par l'ACO

Comme le GA l'optimisation par l'ACO commence par l'étude du modèle CAO et l'extraction de ses données pour les exploiter dans la phase d'optimisation des SD. La phase d'optimisation commence par la construction des SD en se basant sur les matrices de collision en trois directions (x, y, z). Les matrices utilisées sont présentées dans la Figure 34.

$$[I_x] = \begin{pmatrix} 0&0&0&0&0&0&1&1&1&1&0&0&0&0&0&0 \\ 1&0&0&1&1&1&1&1&1&0&0&0&0&0&1&1 \\ 1&1&0&1&1&1&0&0&1&1&1&1&1&1&1&1 \\ 1&1&1&0&0&1&1&1&0&0&0&0&0&0&1&1 \\ 1&1&1&1&0&1&1&1&0&0&0&0&0&0&1&1 \\ 1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 1&0&0&0&0&0&0&0&1&0&1&0&0&0&0&0 \\ 1&0&0&0&0&0&0&0&1&1&1&0&0&0&0&0 \\ 1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 1&0&0&0&0&1&0&0&0&0&0&0&0&0&1&0 \end{pmatrix} \quad [I_y] = \begin{pmatrix} 0&0&0&0&0&1&1&1&1&1&0&0&0&0&1&1 \\ 0&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0 \\ 1&1&0&1&1&1&1&1&1&1&1&1&1&1&0&1 \\ 0&1&0&0&0&0&1&1&0&0&0&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 1&1&0&0&0&0&0&0&0&0&0&0&0&0&1&1 \\ 1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 1&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0 \\ 1&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0 \\ 0&0&1&0&0&0&0&0&1&0&0&0&0&1&0&0 \\ 0&0&1&0&0&0&0&0&0&1&0&0&1&0&0&0 \\ 0&0&1&0&0&0&0&0&0&0&0&1&0&0&0&0 \\ 0&0&1&0&0&0&0&0&0&0&1&0&0&0&0&0 \\ 1&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0 \\ 1&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0 \end{pmatrix} \quad [I_z] = \begin{pmatrix} 0&0&1&0&0&1&1&1&1&1&1&0&0&1&1&1 \\ 0&0&1&1&1&0&1&1&1&0&1&0&0&1&0&0 \\ 1&1&0&1&1&1&0&0&0&0&1&1&1&1&1&1 \\ 0&1&1&0&0&0&0&1&0&0&1&0&0&1&0&0 \\ 0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 1&1&1&0&0&0&0&1&1&0&1&0&0&1&1&1 \\ 1&1&1&1&0&1&0&1&1&0&1&0&0&1&1&1 \\ 1&1&1&0&0&0&0&0&1&0&1&0&0&1&0&0 \\ 1&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0 \\ 1&0&0&0&0&1&1&1&1&0&0&0&0&0&1&1 \\ 0&0&1&0&0&0&0&0&0&0&0&0&0&1&0&0 \\ 1&1&1&1&0&1&1&1&1&1&1&1&1&1&1&1 \\ 1&1&1&1&0&1&1&1&0&0&1&1&0&1&0&1 \\ 0&0&1&0&0&0&0&0&0&0&1&0&0&0&0&0 \\ 1&0&1&0&0&1&1&1&1&1&1&0&0&0&0&0 \\ 1&1&1&0&0&1&0&1&1&0&1&0&0&1&0&0 \end{pmatrix}$$

Figure 34: Les matrices de collision suivant les trois directions de l'étrier de frein

L'exploitation de ces matrices est présentée dans l'organigramme de la Figure 35. L'initialisation des paramètres de l'ACO commence par deux étapes principales. Tout d'abord, il s'agit de déterminer le nombre de fourmis, noté m, qui seront utilisées pour la recherche de solutions. Il est important de noter que ce nombre n'a pas un impact critique sur la convergence globale de l'algorithme, mais il est plutôt destiné à affiner la solution. Ensuite, le deuxième paramètre clé à initialiser est le coefficient d'évaporation, noté ρ. Ce coefficient reflète la volatilité de la phéromone sur les chemins. Il est essentiel de trouver un équilibre, car une valeur trop élevée peut entraîner le piégeage des fourmis dans des minimas locaux. Par conséquent, il est généralement recommandé d'utiliser une faible valeur, typiquement 0.1, pour éviter ce problème. Le dernier paramètre à prendre en compte lors de l'initialisation est le taux de phéromone initial, τ0. Au début de la recherche, une quantité constante de phéromone est attribuée à chaque arête, qui représente le chemin entre deux pièces dans le contexte de la recherche des SD. Cette quantité initiale de phéromone est répartie uniformément avec une valeur τ0 supérieure à zéro.

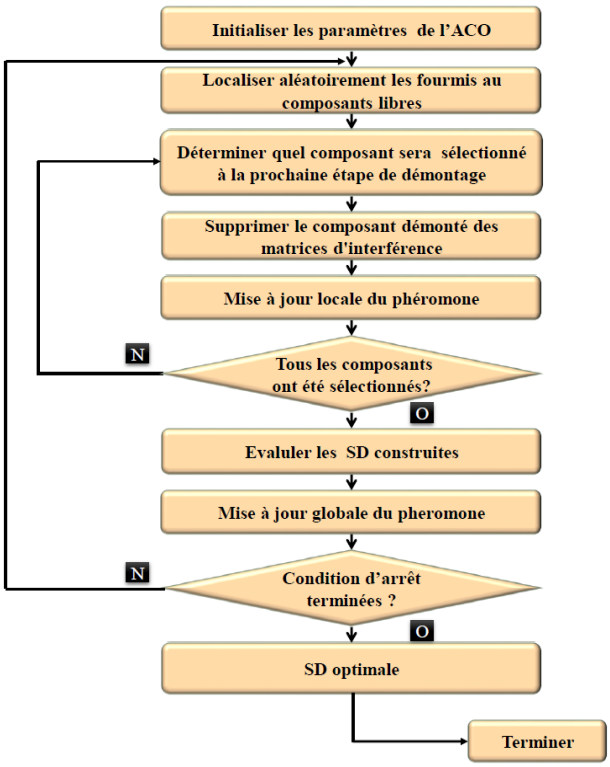


Figure 35: Organigramme de l'algorithme ACO pour la génération des SD (Kheder et al., 2017)

L'un des objectifs de l'approche de Khedher et al. est de créer des SD réalisables en utilisant le concept de Composant Libre (CL). Cette technique implique l'identification des CL à partir des matrices de collision [I_k]. Pour repérer le composant libre (CL (m, + k)) dans la direction de l'axe +k, cette identification s'appuie sur l'utilisation de l'opérateur booléen suivant les deux équations (4) et (5).

Équation 8

$$CL_{m,(+k)} = \bigcup_{j=1}^{N} I_{mj}$$

Équation 9

$$CL_{m,(-k)} = \bigcup_{j=1}^{N} I_{jm}$$

Avec :

- m indique que la pièce Pm est sélectionnée par la fourmi.
- N étant le nombre de composants dans l'assemblage.

U est l'opérateur booléen « ET ». En effet, le résultat de cette opération sera nul lorsque tous les éléments impliqués dans l'opération sont à zéro. Cela signifie que le composant Pm peut être démonté sans aucune contrainte et n'entraînera pas de perturbations avec d'autres composants Pn dans la direction de l'axe + k. C'est ainsi qu'un Composant Libre (CL) est identifié. Parallèlement, pour rechercher les composants à démonter dans la direction opposée (-k), l'équation (10) illustre le processus similaire. Le processus continue jusqu'à la construction de toutes les séquences de démontage réalisables.

Le choix de la séquence optimale est fait en se basant sur une fonction objective (FO) définit comme suit :

Équation 10

$$OF = max\left(N - \left(\frac{\gamma}{D+1} + \frac{\delta}{T} + \frac{\mu}{M} + \psi \times V\right)\right)$$

Avec :

- M est la variable associé à la maintenance telle que définie dans l'Équation 11.

Équation 11

$$M = \sum_{k=1}^{N} \frac{mi}{\sum mi}(N - k + 1)$$

Où : mi représente le facteur relatif à la maintenance pour chaque composant et peut prendre les valeurs suivantes :

  - 1 s'il n'y a pas besoin de maintenance pour le composant i.
  - 2 s'il est nécessaire d'effectuer une maintenance corrective sur le composant i.
  - 3 s'il est nécessaire d'effectuer une maintenance préventive sur le composant i.
- V : est la somme des volumes relatifs de chaque composant de volume V_i dans le mécanisme, donné par l'Équation 12 :

Équation 12

$$V = \sum_{i=1}^{N} \frac{Vi}{\sum Vi}(N - i + 1)$$

- D et T représentent respectivement la valeur totale du nombre de changements de direction

(d) et la valeur totale du nombre de changements d'outil (T) de l'opération(op) de démontage, donnés par :

Équation 13

$$D = \sum_{i=1}^{N-1} d(op_i, op_{i+1})$$

Équation 14

$$T = \sum_{i=1}^{N-1} T(op_i, op_{i+1})$$

- γ, δ, μ et ψ représentent des coefficients de pondération qui peuvent être choisis en fonction des objectifs du concepteur.
  Pour valider l'approche, Khedher et al. ont choisi les coefficients de pondération comme suit : γ = δ = ψ = 0,2 et μ = 0,4.

L'implémentation de l'ACO pour l'optimisation des SD de l'étrier de frein donne les résultats représentés par le Tableau 15:

Tableau 15: SD optimale et ses attributs obtenus par l'ACO

PSD Optimal	7	8	2	4	5	9	10	1	3	13	14	11	12	16	6	15
Direction	+x	+x	+y	+y	+y	+x	+x	+x	-x	-x	-x	-x	-x	-x	-x	-x
Tools	G3	G3	G1	G1	G1	G2	G2	G1	G1	G1	G1	G4	G4	G1	G5	G6

L'optimisation des SD par l'algorithme ACO montre sa faisabilité au niveau de la priorité des pièces concernées par la maintenance ainsi que le nombre de changement d'outil et de direction, comme pour l'algorithme GA.

La section suivante présentera une comparaison des deux méthodes avec notre approche d'optimisation des SD par apprentissage par renforcement.

3.3 COMPARAISON DES RESULTATS ET DISCUSSION

L'optimisation des SD par le GA, l'ACO ou l'apprentissage par renforcement nécessitent la formulation d'une fonction objectif (OF) ou fonction fitness qui intègre les paramètres à optimiser.

Le Tableau 16 présente les 2 fonctions d'optimisation adoptées par Khedher et al. et celle adoptée dans notre approche ainsi que les paramètres pris en compte et les poids considérés lors de l'implémentation informatique.

Tableau 16: Les fonctions d'optimisation adoptées pour chaque approche

Approche	Fonction objective	Poids	Paramètres considérer
Khedher et al. 2015(AG)	$OF = N + \frac{\alpha}{M} + \beta \times D + \gamma \times T + \mu \times V$	α = -0.4 et β= γ = μ= -0.2	Changement d'outil et direction de démontage
Khedher et al. 2017(ACO)	$OF = max\left(N - \left(\frac{\gamma}{D+1} + \frac{\delta}{T} + \frac{\mu}{M} + \psi \times V\right)\right)$	γ= δ = ψ = 0.2 and μ = 0.4.	
Optimisation des SD pour la fin de vie	$F(state, action) = \alpha \times \left(1 - \left(\frac{V_{(i)}}{Vmax}\right)\right) + \beta \times \left(1 - \left(\frac{t_{(i)}}{t_{max}}\right)\right) + \gamma \times D(i) + \delta \times T(i)$	α=β=δ= γ=0.25	Changement d'outil et direction de démontage. Temps de démontage et accès au pièces d'usure et au petite pièces en priorité
Optimisation des SD pour la maintenance	$F(state, action) = \alpha \times \left(1 - \left(\frac{V_{(i)}}{Vmax}\right)\right) + \beta \times \left(1 - \left(\frac{t_{(i)}}{t_{max}}\right)\right) + \gamma \times D(i) + \delta \times T(i) + \lambda P(i)$	α=β=δ= γ=λ=0.2	Changement d'outil et direction de démontage. Temps de démontage et accès au petites pièces en priorité.

Tableau 17: Comparaison des résultats obtenus par : GA, ACO et RL.

Méthode utilisée	SD optimale															
AG	7	8	2	4	5	10	9	12	11	16	13	14	3	1	6	15
Direction	+x	+x	+y	+y	+z	+x	+x	+x	-x	-x	-x	-x	-x	-x	-x	-x
Outil	G3	G3	G1	G1	G1	G2	G2	G1	G1	G1	G1	G4	G4	G1	G5	G6
ACO	7	8	2	4	5	9	10	1	3	13	14	11	12	16	6	15
Direction	+x	+x	+y	+y	+y	+x	+x	+x	-x	-x	-x	-x	-x	-x	-x	-x
Outil	G3	G3	G1	G1	G1	G2	G2	G1	G1	G1	G1	G4	G4	G1	G5	G6
RL DC	7	8	9	10	1	6	15	11	12	13	14	16	2	4	5	3
Outil	G3	G3	G2	G2	G1	G5	G6	G4	G4	G1	G1	G1	G1	G1	G1	G1
Direction	+x	+x	+x	+x	+x	+x	+x	+x	+x	+x	+x	+x	+y	+y	+y	+y
RL DP	7	8	2	4	5	9	10	1	11	12	6	15	13	14	16	3
Tools	G3	G3	G1	G1	G1	G2	G2	G1	G4	G4	G5	G6	G1	G1	G1	G1
Direction	+x	+x	+y	+x	+x	+x	+x	+x	+x	+x	+x	+x	+x	+x	+x	+x

Tableau 18: Comparaison de GA, ACO et RL pour la minimisation du nombre d'outils et de changements de direction.

	Nombre de changement d'outil	Nombre de changement de direction
AG	7	4
ACO	6	3
RL DC (EoL)	6	1
RL DP (MP)	6	2

Pour plus de résultats et de validations de l'approche proposée, la Figure 36 présente une comparaison entre l'apprentissage par renforcement pour le démontage partiel (RL DP) et le démontage complet (DC) avec les deux approches de la littérature (GA et ACO). Pour une meilleure comparaison, le meilleur temps et le meilleur coût de démantèlement lors de l'épisode sont considérés en mettant également l'accent sur la minimisation des changements d'outils et de direction. Suite à la comparaison de la Figure 36 suivante, la méthode proposée prouve son efficacité avec des scénarios de désassemblage moins chers et plus rapides dans les deux cas : RL pour DP et RL pour DC.

Selon le

Tableau 18, l'approche proposée a permis de minimiser le nombre de changements d'outils. En effet, il été de 7 avec le GA et 6 avec l'ACO et est resté à 6 avec l'utilisation de l'algorithme RL. D'autre part, l'approche proposée a permis une réduction du nombre de changements de direction. En

effet, pour l'exemple traité, le changement de direction était de 4 avec le GA et 3 avec l'ACO alors qu'il est seulement de 1 ou 2 changement(s) de direction lors de l'utilisation de RL.

De plus, l'approche proposée permet de gagner du temps de manière avantageuse. Le temps nécessaire pour traiter une SD sur le même PC est de 956 (s) avec le GA et 926 (s) avec l'ACO. Ce temps devient égal à 911 (s) pour le PD et 873 (s) pour le DC lors de l'utilisation de l'algorithme RL.

En ce qui concerne le coût de la SD, l'approche proposée ne prend en compte que la main-d'œuvre sans inclure les coûts liés aux machines. Ainsi, pour une main-d'œuvre à 30 $/heure, le coût d'une SD de 16 pièces est passé de 8$ pour l'AG et 7,7$ pour l'ACO à 7$ pour l'algorithme RL DP et de 6$ pour le RL DC.

Par ailleurs, l'optimisation par l'algorithme RL nécessite un temps d'exécution d'une seconde pour générer la SD optimale. Pour cela, l'algorithme RL nécessite une bonne configuration des paramètres. Nous citons dans ce qui suit quelques points importants pour la configuration de l'algorithme RL :

- Le volume des données est le principal facteur influençant le temps d'exécution.
- Les réglages de l'algorithme RL (par exemple la variation du facteur d'actualisation), peuvent affecter la convergence de l'algorithme, c'est pourquoi dans cet algorithme, le facteur d'actualisation est toujours fixé à 0,8.
- Le niveau d'ajustement (par exemple le temps d'exécution sans utiliser la fonction fitness), est de 0.5 seconde, mais après l'ajout de la fonction fitness, le temps d'exécution est multiplié par 2.
- Les caractéristiques matérielles. Cet algorithme a été mis en œuvre sur un ordinateur avec un processeur Intel Core i5-11400H de la 11eme génération à 2,70 GHz, avec 16,0 Go (15,7 Go utilisables) de RAM. Augmenter ou diminuer les capacités de l'ordinateur peut entraîner des augmentations ou des diminutions consécutives du temps d'exécution.

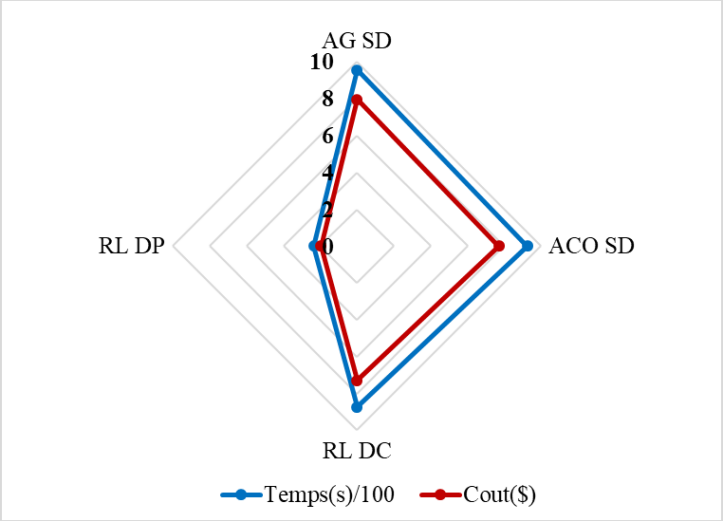


Figure 36: Comparaison des coûts et des temps pour les trois algorithmes : GA, ACO, RL.

La comparaison effectuée ci-dessus démontre l'efficacité de la méthode proposée pour optimiser simultanément 5 paramètres de la SD en seulement 1 seconde d'exécution. Cela contribue à réduire le temps et les coûts de l'exécution d'une SD.

La ré-implémentation des algorithmes existants pour extraire une SD optimale de l'étrier de frein a donné des temps d'exécution informatique relativement proches pour les trois approches d'optimisation (GA, ACO et RL) (Figure 37). Pour extraire la SD optimale, l'ACO consomme 1.8 (s) de temps d'exécution, le GA 1.3 (s) et le RL seulement une seconde. L'approche RL reste donc plus avantageuse en termes de SD optimale avec un minimum de temps d'exécution et de réalisation de la séquence par rapport aux deux autres approches, ce qui démontre non seulement la faisabilité mais aussi l'efficacité de l'approche proposée.

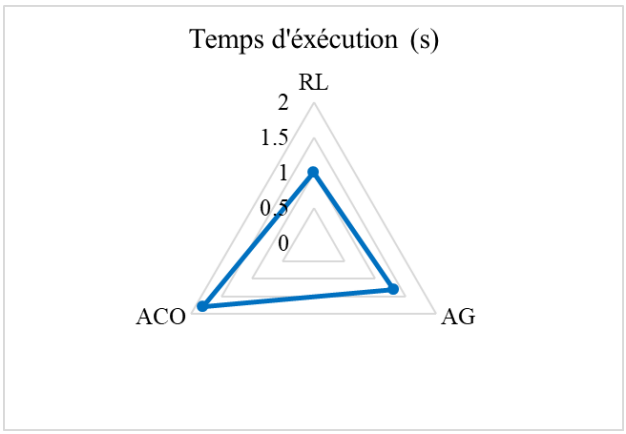


Figure 37: Comparaison du temps d'exécution des trois algorithmes d'optimisation : GA, ACO et RL

4 CONCLUSION

Ce chapitre présente une comparaison entre les résultats de travaux issus de la littérature et l'approche proposée dans le cadre de cette thèse. Dans un premier temps, le choix des outils d'implémentation informatique de l'approche proposée sont introduits. Un exemple de validation tiré de la littérature est ensuite présenté. Les deux travaux issus de la littérature sont alors présentés avec leurs résultats. Une comparaison entre deux travaux basés sur le GA et l'ACO et l'approche proposée basée sur l'apprentissage par renforcement est ensuite exposée. Cette comparaison a finalement montré l'efficacité de l'approche proposée. La méthode d'optimisation proposée est appliquée à deux stratégies de démontage selon le souhait de l'utilisateur : (1) la planification des SD pour le démontage partiel (PSD DP) ; (2) la planification des SD pour le processus de démontage complet en fin de vie (PSD DC). Le processus d'optimisation intègre cinq paramètres, à savoir : (1) la minimisation des changements d'outil de démontage ; (2) la minimisation des changements de direction de démontage ; (3) l'optimisation du temps de démontage, y compris le temps de préparation et de traitement, ainsi

que deux paramètres de pratiques industrielles : (4) la priorisation du démontage des pièces les plus petites ; et (5) la facilité de l'accès aux pièces d'usure. Comparée à l'approche GA, l'approche RL permet d'économiser 5 % de temps pour une SD partielle et 9 % de temps pour une SD complète. De plus, comparée à l'ACO, l'approche RL permet d'économiser 2 % de temps pour une SD partielle et 6 % de temps pour une SD complète.

Le chapitre suivant présente la reconception des assemblages, pour l'optimisation du processus d'assemblage et de désassemblage.

CHAPITRE 4 : RECONCEPTION POUR L'OPTIMISATION DES PROCESSUS D'ASSEMBLAGE ET DE DESASSEMBLAGE

1 INTRODUCTION

L'optimisation des séquences d'assemblage et de désassemblage (SA/SD) ne se limite pas aux paramètres de réalisation de leur séquence ni au traitement algorithmique et temps d'exécution pour extraire la séquence optimale. L'architecture du modèle CAO et le niveau de complexité du système ont également une influence considérable sur la planification des SA/SD partielle ou complète. Ce chapitre présente une approche d'optimisation des SA/SD par la reconception d'une architecture existante et la création d'un modèle CAO simplifié principalement en terme du nombre de pièces. L'objectif de cette reconception est de faciliter l'accès à certaines pièces d'usure qui nécessitent une maintenance à haute fréquence. La première partie de ce chapitre détaille le problème de reconception pour l'optimisation des SA/SD. La deuxième partie de ce chapitre présente l'approche proposée. La dernière partie de ce chapitre présente la validation de l'approche proposée sur deux exemples un robot TurtleBot3 Burger et un moteur de manutention.

2 OPTIMISATION DES PROCESSUS D'ASSEMBLAGE ET DE DESASSEMBLAGE PAR RECONCEPTION DE LA SOLUTION PROPOSEE

De nos jours, les assemblages sont bien instrumentés avec des pièces intelligentes telles que des capteurs intégrés dans les pièces, des microcontrôleurs et des cartes Raspberry Pi, etc. Ces éléments sont utilisés pour surveiller et transmettre les données du système en temps réel. Plus que la qualité et l'efficacité, les exigences des clients ont contribué à l'augmentation de la complexité de l'architecture CAO des produits, leur coût et leur processus de maintenance.

Les deux chapitres précédents ont présenté l'optimisation des SD en agissant sur des paramètres industriels liés directement à la procédure de réalisation de la SD elle-même, ainsi que des paramètres liés au temps CPU d'exécution pour la générer. Cela a mené à des résultats intéressants qui peuvent être améliorés à travers l'amélioration de la conception CAO de l'assemblage.

Dans le cadre de l'amélioration de la conception, une méthode a été développée : la conception pour la fabrication. Cette méthode consiste à améliorer les conceptions afin de les fabriquer sous la forme d'une nouvelle génération de produits (Herrmann et al., 2004). Cette méthode est expliquée à travers l'exemple démonstratif de principe de conception pour la fabrication : un moteur de manutention composé de 18 pièces (Figure 38). Le moteur (3) est fixé à la base en aluminium (1) par les deux vis de fixation Ø 0.2×0.6 (4). Le capteur de position (5) est fixé au logement de la base en aluminium (1) par les deux écrous Ø 0.06×0.12 (12). Le moteur est protégé par le couvercle (10) fixé par 4 vis, deux vis (7) sur la base et deux vis Ø 0.12×0.3 (11) sur la plaque de fermeture (8). L'écart entre la base en

aluminium (1) et la plaque de fermeture (8) est réalisé par les deux écarteurs (6). Les bagues (2) et (9) sont utilisées pour le serrage. Le

Tableau 19 illustre la nomenclature des composants du moteur de manutention.

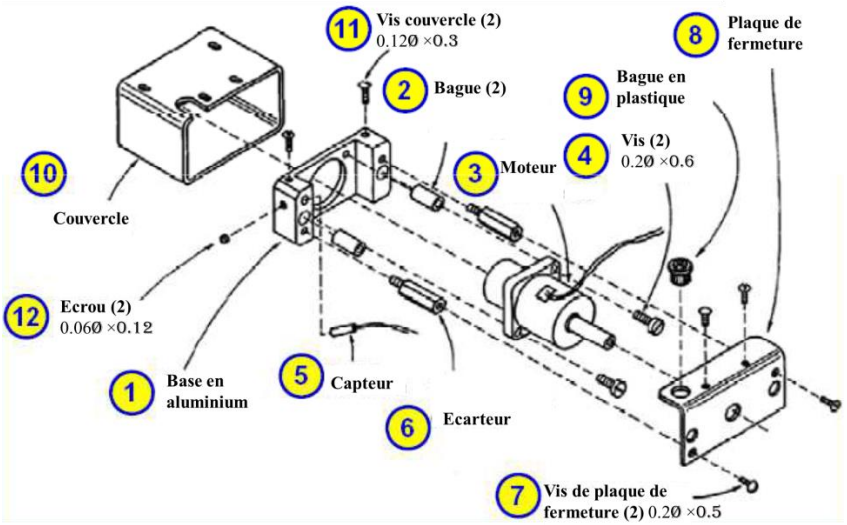


Figure 38: Modèle CAO du moteur[3]

Tableau 19: Nomenclature des composants du moteur

Composants	Nomenclature	Nombre
1	Base en aluminium	1
2	Bague	2
3	Moteur	1
4	Vis	2
5	Capteur	1
6	Ecarteur	2
7	Vis de plaque de fermeture	2
8	Plaque de fermeture	1
9	Bague en plastique	1
10	Couvercle	1
11	Vis de couvercle	2

[3] https://www.dfma.com/

12	Ecrou	2

Le principe de conception pour la fabrication est de réduire le nombre de composants dans l'ensemble de l'architecture CAO, cette minimisation de composant est basée sur 4 questions :

- La pièce est-elle une base ?
- Existe-t-il un mouvement relatif fonctionnel entre la pièce et les autres pièces déjà assemblées ?
- La pièce doit-elle être de matériau différent ou isolée des autres pièces ?
- La pièce doit-elle être séparée des autres pièces pour faciliter l'assemblage ?

Après minimisation du nombre de composants comme le montre le Tableau 20, la nouvelle conception subit un calcul d'efficacité pour la comparer avec la conception initiale.

Tableau 20 : Identification des possibilités de suppression des pièces

Numéro	Nomenclature	Description	Nécessaires	Pas nécessaire
1	Base en aluminium	1ère pièce à positionner ne peut pas être combinée	+	
2	Bague	Peuvent être de même matière que la base		+
3	Moteur	Sous-assemblage standard	+	
4	Vis	Un autre arrangement est possible		+
5	Capteur	Sous-assemblage standard	+	
6	Ecarteur	Peuvent s'intégrer à la base		+
7	Vis de plaque de fermeture	Doit être une pièce séparée	+	
8	Plaque de fermeture	Un autre arrangement est possible		+
9	Bague en plastique	Peut-être combiner avec la plaque de fermeture		+
10	Couvercle	Peut-être combiner avec la plaque de fermeture		+
11	Vis de couvercle	Supprimé si possible		+
12	Ecrou	Supprimé si possible		+

La nouvelle conception proposée présentée dans la Figure 39 permet d'économiser 9 pièces, par rapport à l'architecture initiale composée de 18 pièces, la nouvelle architecture est composée de 9

pièces.

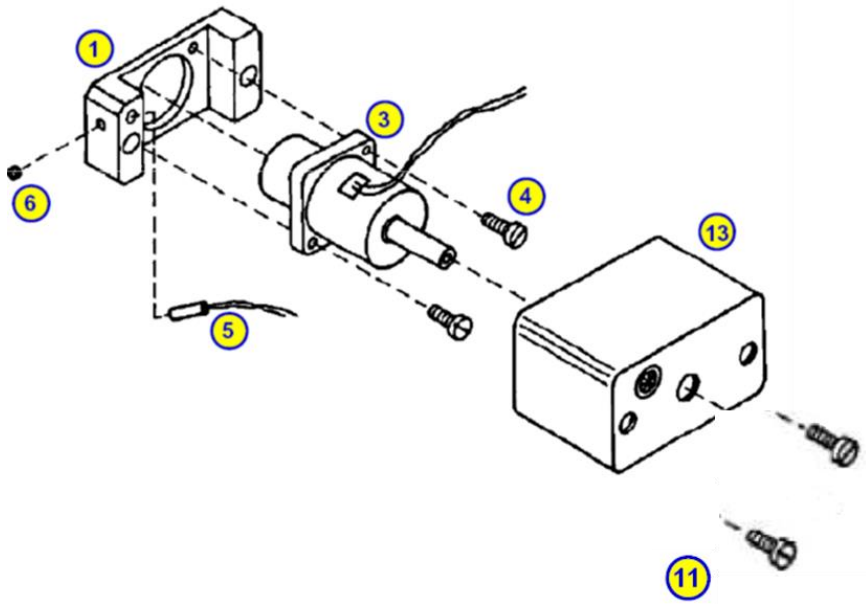


Figure 39: Nouvelle conception proposée

Tableau 21: Nouvelle composition du moteur

Composants	Nomenclature	Nombre
1	Base en aluminium	1
3	Moteur	1
4	Vis	2
5	Capteur	1
6	Ecrou	1
11	Vis de couvercle	2
13	Couvercle	1

En examinant la nouvelle conception d'un point de vue assemblage et désassemblage sélectif et complet, les SA/SD complètes initiales du bloc moteur sont présentées dans le Tableau 22.

Tableau 22: SA/SD complètes optimales

Ordre	SA initiale	SA optimale	SD initiale	SD optimale
1	Base en aluminium	Base (1)	Vis de plaque de fermeture	2 vis couvercle (11)
2	2 bagues	Ecrou(6)	Vis de couvercle	Couvercle (13)
3	2 Ecarteurs	Capteur(5)	Bague en plastique	2 vis(4)
4	Ecrou couvercle	Moteur(3)	Couvercle	Moteur(3)
5	Capteur	2 vis(4)	Plaque de fermeture	Capteur(5)
6	Moteur	Couvercle (13)	2 vis	Ecrou(6
7	2 vis	2 vis couvercle (11)	Moteur	
8	Plaque de fermeture		Capteur	
9	Couvercle		Ecrou couvercle	
10	Bague en plastique		2 Ecarteurs	
11	Vis de couvercle		2 bagues	
12	Vis de plaque de fermeture		Base en aluminium	

Pour la SA et la SD sélective, le but est de changer le moteur en cas d'endommagement. Le Tableau 23 présente les SA/SD pour changer le moteur. Une opération de changement du moteur nécessite initialement 7 étapes pour arriver à accéder au moteur :

1. Le desserrage de la vis de la plaque de fermeture.
2. Le desserrage de la vis de couvercle.
3. Le désassemblage de la bague en plastique.
4. Le désassemblage du couvercle.
5. Le désassemblage de plaque de fermeture.
6. Le desserrage des deux vis de fixation du moteur sur la base en aluminium.
7. Le désassemblage de moteur.

Une fois le moteur changé ; il faut faire le même travail pour assembler le moteur comme le montre le Tableau 23.

Tableau 23: Séquence d'assemblage et de désassemblage sélectif du moteur

Ordre	SA sélective initiale	SD sélective initiale
1	Moteur	Vis de plaque de fermeture
2	2 vis	Vis de couvercle
3	Plaque de fermeture	Bague en plastique
4	Couvercle	Couvercle
5	Bague en plastique	Plaque de fermeture
6	Vis de couvercle	2 vis
7	Vis de plaque de fermeture	Moteur

Une opération de maintenance ou de changement du moteur initial est faite en 7 étapes. Grâce à la nouvelle conception une SD du moteur est maintenant réalisable sur 4 étapes uniquement (Tableau 24) :

- Desserrage des 2 vis du couvercle.
- Désassemblage du couvercle.
- Desserrage des 2 vis de fixation du moteur sur la base.
- Désassemblage du moteur.

Tableau 24: SA/SD sélectives optimales

Ordre	SA sélective initiale	SA sélective optimale	SD sélective initiale	SD sélective optimale
1	Moteur	Moteur(3)	Vis de plaque de fermeture	2 vis couvercle (11)
2	2 vis	2 vis(4)	Vis de couvercle	Couvercle (13)
3	Plaque de fermeture	Couvercle (13)	Bague en plastique	2 vis (4)
4	Couvercle	2 vis couvercle (11)	Couvercle	Moteur(3)
5	Bague en plastique		Plaque de fermeture	
6	Vis de couvercle		2 vis	
7	Vis de plaque de fermeture		Moteur	

Les résultats encourageants obtenus grâce à la méthode de conception pour la fabrication, nous ont inspirés pour élaborer une nouvelle méthode de conception pour l'assemblage et le désassemblage. Cette nouvelle méthode vise à optimiser les séquences d'assemblage et de désassemblage dès la phase de conception ou même après la fabrication de produit en proposant des nouvelles conceptions optimales pour les prochaines générations des produits.

La section suivante présente l'approche proposée pour l'optimisation de SA/SD à travers l'optimisation du modèle CAO du produit.

Avant de présenter les différentes étapes de l'approche proposée, nous commençons par identifier les principales hypothèses de modélisation adoptées ainsi que les exemples de validation.

2.1 HYPOTHESES DE MODELISATION

- **Hypothèse 1 :** L'approche proposée couvre le cas de reconception d'un produit existant.
- **Hypothèse2 :** Les pièces de fixation sont prises en compte dans la SA/SD.
- **Hypothèse 3 :** Le mécanisme reste obligatoirement fonctionnel même après la reconception.

2.2 PRESENTATION DE L'EXEMPLE D'APPLICATION

L'approche proposée a été validée sur un robot éducatif, le TurtleBot3 Burger. Il s'agit d'un robot connecté. Ce robot est utilisé dans des missions de cartographie grâce à son capteur LiDAR 360°. Ce capteur émet des faisceaux lasers pour mesurer les distances vers les objets autour de lui. Tout d'abord, le capteur LDS-01 est monté sur le robot, et envoie des faisceaux lumineux pour calculer les distances entre le robot et les objets à proximité. Ce capteur tourne pendant cette opération pour balayer l'environnement à 360 degrés, en enregistrant en continu les mesures de distance. En résumé, le TurtleBot3 Burger utilise son capteur LDS-01 pour cartographier son environnement, ce qui lui permet de se localiser et de naviguer de manière autonome en utilisant les cartes résultantes. Ces cartes enregistrées sont essentielles pour son déplacement intelligent et sa réactivité aux changements environnementaux. Ce capteur représente donc la partie la plus importante du robot. Le TurtleBot3 Burger est connecté et manipulé par la plateforme Robot Operating System (ROS) où il peut envoyer des informations en temps réel et afficher les cartes qui ont été générées. Ainsi l'endommagement du LiDAR (LDS-01) désactive toutes les fonctionnalités du robot. Cette partie doit donc être facile à maintenir en cas de panne. Cette étude sera basée sur le processus de maintenance du LDS-01 avec l'application de la méthode de reconception pour l'optimisation des SA/SD. Le capteur LDS-01 est fixé au système à l'aide de 4 vis M2.5*16mm, 4 entretoises, 4 supports de PCB, 4 écrous M2.5, 4 vis M2.5*8mm, câble électrique du LDS-01 et le capteur LDS-01. Tous ces composants sont fixés à 2 plaques (waffle-Plate) qui sont assemblées par 4 vis M3*8mm et 4 écrous M3.

La Figure 40 illustre le modèle CAO du robot.

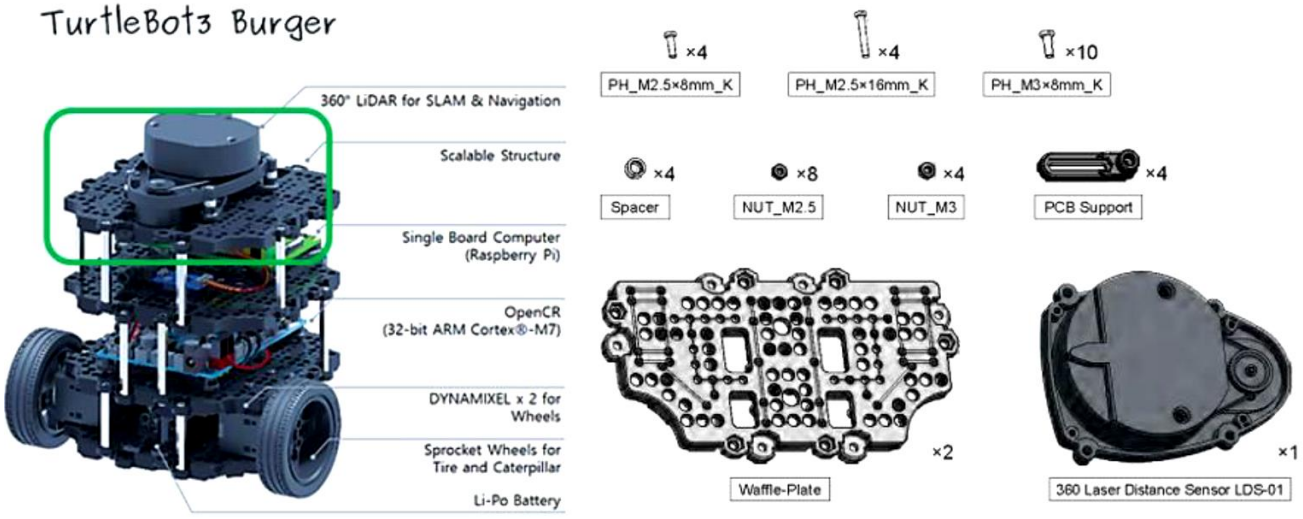


Figure 40: TurtleBot3 burger

Le **Error! Not a valid bookmark self-reference.** présente la nomenclature des différents composants du robot.

Tableau 25: Nomenclature des composants du TurtleBot 3 Burger

Composants	Nomenclature	Nbre	Composants	Nomenclature	Nbre
Pièce de châssis	Plaques	8		Câble d'alimentation	1
	Support plat M3*35mm	4		Batterie en lithium 11.1V 1,800mAh	1
	Support plat M3*45mm	10		Chargeur de batterie	1
	Support PCB	12	Outils	Tournevis	1
	Roue	2		Outil à riveter	1
	Pneu	2	Divers	Vis_M2x4mm_K	8
	Boule	1		Vis_T2x6mm_K	4
Moteur	DYNAMIXEL (XL430-W250-T)	2		Vis_M2.5x8mm_K	16
cartes	OpenCR1.0	1		Vis_T2.6x12mm_K	16
	Raspberry Pi	1		Vis_ _M2.5x16mm_K	4
	USB2LDS	1		Vis_ M3x8mm_K	44
Capteur	LDS-01	1		Ecrou_M2.5	20
Carte mémoire	Carte Micros	1		Ecrou_M3	16
Câbles	Câble d'alimentation de Raspberry Pi	1		Rivet_1	14
	Câble d'extension pour batterie	1		Rivet_2	2
	Câble de lien entre DYNAMIXEL et Open CR	2		Entretoise	4
	Câble USB	2		Support	5

Alimentation	SMPS 12V5A	1	Plaque d'adaptation	1

Après la description de l'exemple choisi, la section suivante présente l'approche proposée pour l'optimisation de SA/SD par la reconception de produits existants.

3 LA RECONCEPTION POUR L'OPTIMISATION DES PROCESSUS D'ASSEMBLAGE ET DE DESASSEMBLAGE

Le cycle de vie d'un produit mécanique est composé de plusieurs phases distinctes, débutant par sa conception initiale et s'étendant jusqu'à sa fin de vie. La Figure 41 présente une synthèse du cycle de vie d'un produit mécanique ainsi que l'étendue de l'application de notre approche. Dans ce qui suit, nous détaillons les différentes étapes de l'approche proposée.

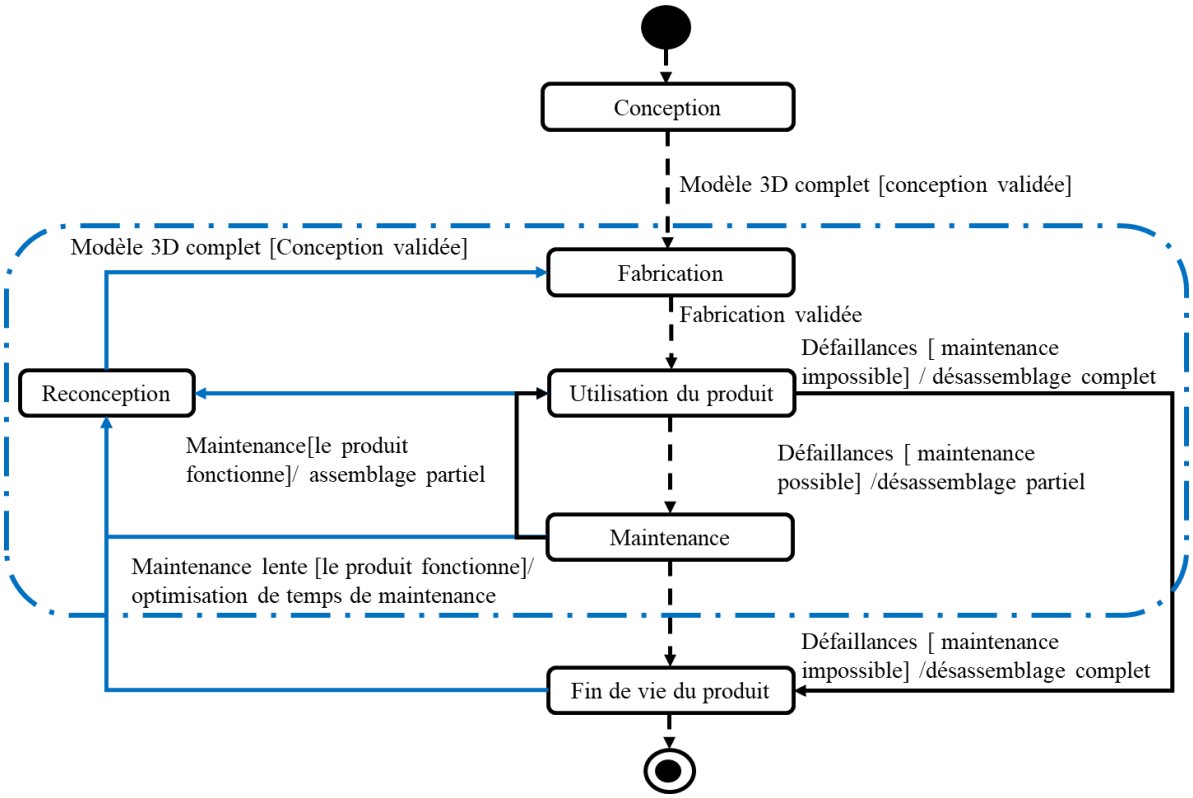


Figure 41: Phases du cycle de vie impliquée dans l'approche proposée dans le cycle de vie du produit

- **La phase de conception** : Au début du cycle, le produit est imaginé en tenant compte des exigences du marché et des spécifications techniques. Cette étape comprend la création de schémas, de modèles 3D et de prototypes pour donner naissance au produit.
- **La phase de fabrication** : Après la phase de conception, le produit est industrialisé et fabriqué à grande échelle. Cette étape peut impliquer la production des composants, l'assemblage et le

contrôle qualité.

- **La phase d'utilisation** : En cours d'utilisation, le produit mécanique peut avoir des défaillances et des interventions pour la maintenance et la réparation.
- **La phase de fin de vie** : Enfin, dans la phase de fin de vie, le produit peut être complètement désassemblé, recyclé ou éliminé dans le respect des normes environnementales.

Après la conception, la fabrication, l'utilisation et la défaillance du produit, l'approche proposée utilise le principe de reconception pour l'assemblage, et met en évidence le retour d'expérience de la maintenance pour identifier et corriger les problèmes observés. Ce retour d'expérience peut être exprimé en termes de difficulté d'accès de la maintenance à certaines pièces, ce qui augmente le coût et le temps de la maintenance.

La démarche de reconception de l'assemblage ou du désassemblage consiste à étudier ces problèmes et à proposer une nouvelle conception optimale du produit. L'architecture reconçue doit satisfaire l'aspect fonctionnel du premier produit, tout en minimisant les coûts et le nombre de pièces assemblées. La nouvelle conception doit garantir une nouvelle séquence optimale d'assemblage ou de désassemblage. Une fois la nouvelle conception est validée, elle sera fabriquée comme une nouvelle génération de produits. La Figure 42 présente les étapes de l'approche de manière plus détaillée :

- Initialisation des entrées :

Les données d'entrée sont : l'architecture du mécanisme et les problèmes trouvés lors de la maintenance. Ces problèmes sont spécifiés selon le cas de la maintenance effectuée, ils peuvent être définir comme suit :

- Une SD partielle complexe en termes d'accès aux pièces d'usures.
- Une grande fréquence de changements d'outil et de direction lors de la réalisation de la séquence.
- Des pièces manquantes dans le marché.
- Un problème de manipulation de l'assemblage lors de la maintenance.
- Architecture CAO complexe etc.

- Etude de la structure CAO :

Ce travail aborde l'architecture de l'assemblage, le type de matériaux utilisés pour chaque pièce, le type de fixations et les types de contraintes mécaniques entre les pièces. Cette étude est réalisée pour trouver toutes les combinaisons possibles de pièces. Cette combinaison n'est pas seulement géométrique, mais affecte également le type de matériaux utilisés pour les pièces, c'est-à-dire que si deux pièces sont faites de matériaux différents, leur combinaison directe est interdite. Quand une combinaison entre deux pièces est possible, il est nécessaire de déterminer l'efficacité de cette combinaison au regard de la résistance des pièces et de l'analyse globale de la nouvelle structure.

➢ Combinaison des pièces (CP) :

- La combinaison des pièces est basée sur l'étude de l'architecture détaillée ci-dessous. S'il est possible de combiner la pièce i avec la pièce i+1, la nouvelle conception sera proposée pour validation. Sinon, la possibilité de combinaison de la pièce (i, i+1) sera ignorée et ce processus se répète jusqu'au traitement de toutes les possibilités de combinaison.
- La possibilité de combinaison commence par le test du matériau de chaque couple de pièces successivement.
    - M (i, i+1) désigne le matériau de deux pièces étudiées :
        - M (i, i+1) =1 si les des deux pièces étudiées sont de même matériau.
        - M (i, i+1) =0 si les des deux pièces étudiées sont de matériau différent.
    - C (i, i+1) désigne le contact entre les deux pièces étudiées :
        - C (i, i+1) =1 s'il y a un contact entre les deux pièces (i, i+1).
        - C (i, i+1) =0 sinon.

Équation 15 :

$$\boldsymbol{CP_{i,i+1} = M_{i,i+1} + C_{i,i+1}}$$

Si CP (i, i+1) = 2 (voir Équation 15) la combinaison est réalisable, sinon les deux pièces seront ignorées. Finalement la nouvelle conception optimisée sera proposée pour validation et fabrication.

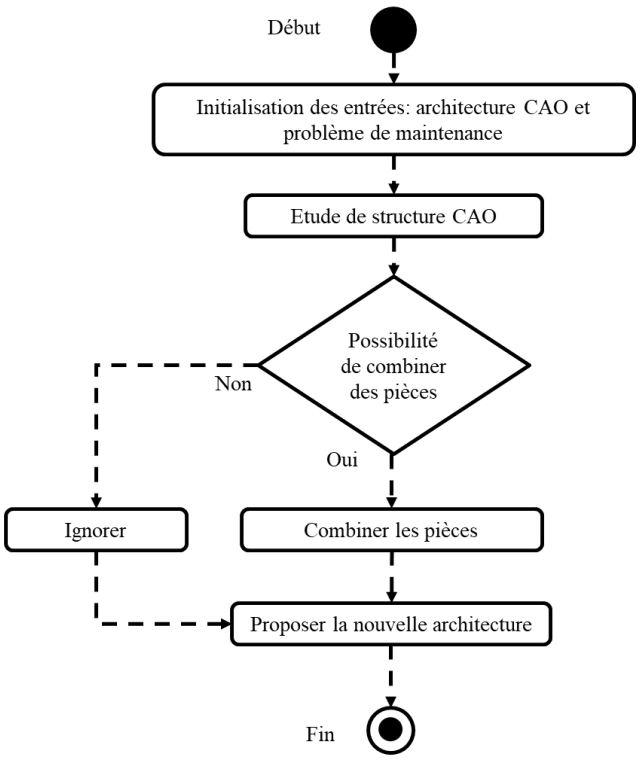


Figure 42: Conception pour l'optimisation de SA/SD

4 VALIDATION DE L'APPROCHE PROPOSEE : TURTLEBOT3 BURGER

Le TurtleBot3 Burger est un robot vendu démonté, accompagné d'un guide indiquant toutes les étapes de l'assemblage. L'assemblage manuel complet prend 3 heures. Le TurtleBot3 Burger est équipé du capteur LDS-01, dont la fonction principale est de tracer les cartographies industrielles. La défaillance de ce capteur entraînera la défaillance de la fonction principale du TurtleBot. Pour cela, il sera nécessaire d'optimiser ses SA/SD.

L'optimisation de SA/SD dans cet exemple est basée sur la facilité d'accès à la pièce principale du système : le capteur LDS-01. La plupart des pièces du robot Turtlebot3 sont fabriquées à partir du même matériau (plastique). Ces pièces sont fabriquées par injection plastique.

La Figure 43 présente la première étape de l'assemblage du quatrième étage du TurtleBot3 Burger. Les deux plaques sont prises comme déjà assemblées, la fixation des supports PCB sur les plaques est divisée en 3 étapes pour chaque support :

1. L'insertion d'un écrou M2.5 dans le support PCB (entouré en rouge dans la Figure 43).
2. La mise en place du support PCB sur les plaques.
3. La fixation du support PCB avec une vis et un écrou.

Ces étapes sont répétées 4 fois pour fixer les quatre supports PCB.

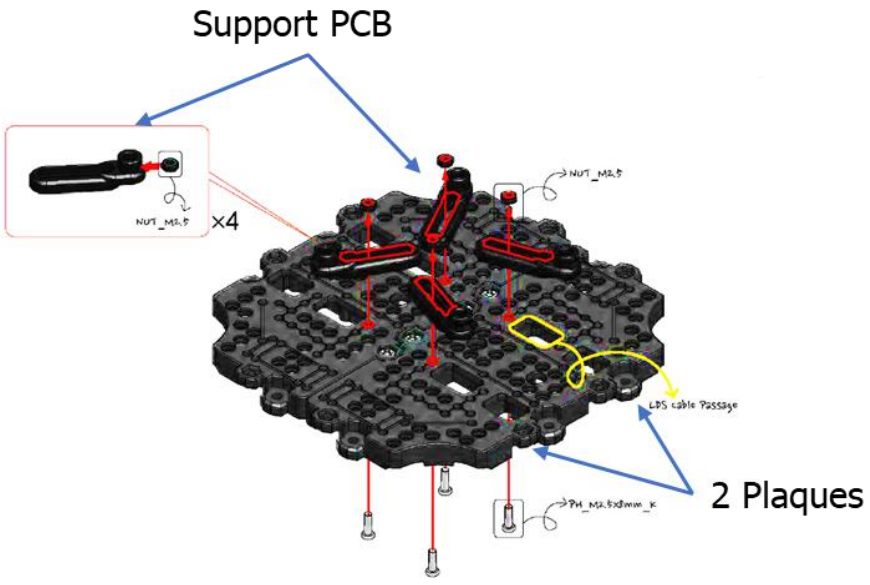


Figure 43: Assemblage des supports PCB sur les plaques du robot

Une fois les Supports PCB sont assemblés avec les plaques. La deuxième étape est présentée dans la Figure 44.

Elle est répartie sur trois étapes :

1. Mise en place des bagues.
2. Mise en place du capteur LDS-01.
3. Fixation des 4 Vis avec les 4 écrou déjà insérer dans les supports PCB.

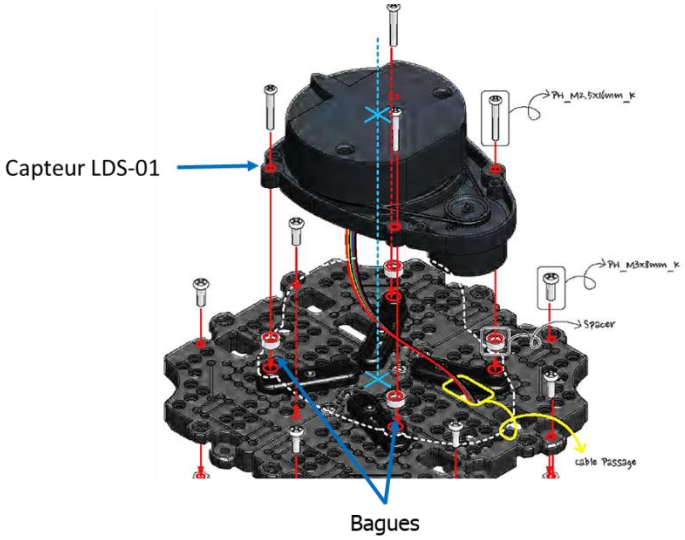


Figure 44: Architecture initiale de l'assemblage du 4ème étage du Turtle Bot3 Burger

Le Tableau 26 représente les détails et l'ordre d'assemblage et de désassemblage des composants afin d'assurer la bonne fixation du capteur LDS-01. Pour ce robot, la SD est l'inverse de la SA. Le désassemblage dépend du cas de la maintenance demandée (changement de Capteur LDS-01, changement d'une des bagues, changement du support PCB). Ces SA et SD sont détaillées dans le Tableau 26.

Tableau 26: SA et SD du capteur LDS-01

Ordre	SA initiale	SD initiale
1	4 Ecrous_M2.5	4 Vis_ M2.5x16mm_K
2	4 supports PCB	Capteur LDS-01
3	Support PCB	4 bagues
4	Vis_ M2.5x8mm_K	Vis_ M2.5x8mm_K
5	Ecrou_M2.5	Ecrou_M2.5
6	Support PCB	Support PCB
7	Ecrou_M2.5	Ecrou_M2.5
8	Vis_ M2.5x8mm_K	Vis_ M2.5x8mm_K
9	Support PCB	Support PCB
10	Ecrou_M2.5	Ecrou_M2.5
11	Vis_ M2.5x8mm_K	Vis_ M2.5x8mm_K
12	Support PCB	Support PCB
13	Ecrou_M2.5	Vis_ M2.5x8mm_K
14	Vis_ M2.5x8mm_K	Ecrou_M2.5
15	4 bagues	Support PCB
16	Capteur LDS-01	4 Ecrous_M2.5
17	4 Vis_ M2.5x16mm_K	4 Supports PCB

Après l'analyse de l'architecture de l'assemblage, l'application de l'approche proposée a donné les informations présentées dans le Tableau 27.

Tableau 27: Identification des possibilités de fusion des pièces

Composant	Description	Nécessaires	Pas nécessaire
8 Ecrous_M2.5	Un autre arrangement est possible		+
4 Vis_ M2.5x8mm	Un autre arrangement est possible		+
Capteur LDS-01	Sous-ensemble standard	+	
4 bagues	Un autre arrangement est possible		+
4 supports PCB	Peut-être fusionner avec les plaques		+
4 Vis_ M2.5x16mm	Un autre arrangement est possible	+	

La nouvelle architecture est présentée dans la Figure 45.

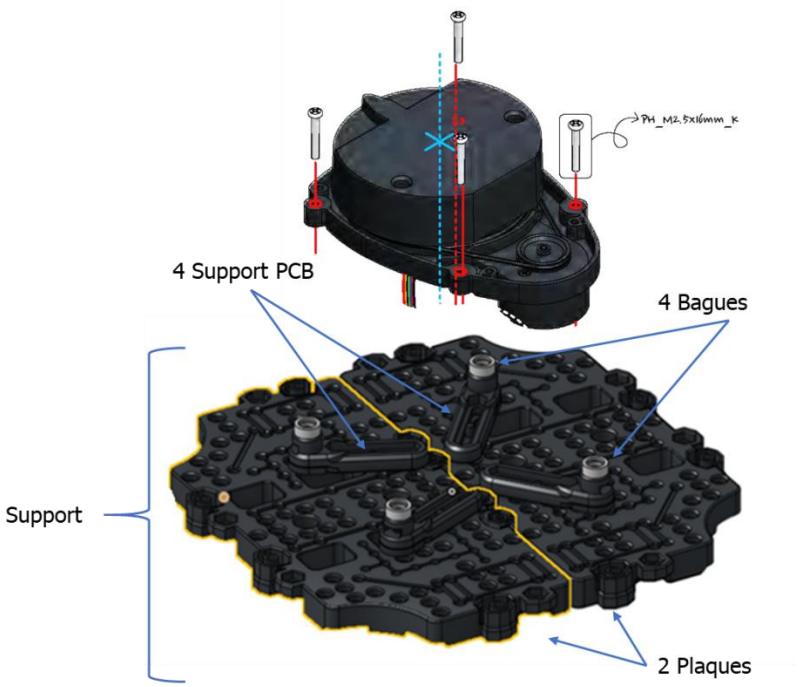


Figure 45: Nouvelle conception proposée

Le nombre de pièces de fixation a diminué de la moitié, suite à la proposition de la fusion des supports PCB et des bagues avec la plaques en une seule pièce. Cette proposition a réduit significativement les SA/SD. Après une SA initiale de 25 pièces, nous passons à une séquence de 10 pièces et le reste est fusionné en une seule pièce « Support ».

Tableau 28: SA/SD après reconception du produit

Ordre	SA initiale	SA optimale	SD initiale	SD optimale
1	4 Ecrous_M2.5	Support	4 Vis_ M2.5x16mm	4 Vis_ M2.5x16mm
2	4 supports PCB	Capteur LDS-01	Capteur LDS-01	4 Ecrou_M2.5
3	Support PCB	4 Vis_ M2.5x16mm	4 bagues	Capteur LDS-01
4	Vis_ M2.5x8mm	4 Ecrou_M2.5	Vis_ M2.5x8mm	Support
5	Ecrou_M2.5		Ecrou_M2.5	
6	Support PCB		Support PCB	
7	Ecrou_M2.5		Ecrou_M2.5	
8	Vis_ M2.5x8mm		Vis_ M2.5x8mm	
9	Support PCB		Support PCB	
10	Ecrou_M2.5		Ecrou_M2.5	
11	Vis_ M2.5x8mm		Vis_ M2.5x8mm	
12	Support PCB		Support PCB	
13	Ecrou_M2.5		Vis_ M2.5x8mm	
14	Vis_ M2.5x8mm		Ecrou_M2.5	
15	4 bagues		Support PCB	
16	Capteur LDS-01		4 Ecrous_M2.5	
17	4 Vis_ M2.5x16mm		4 Supports PCB	

A travers l'approche proposée, les SA/SD partielles pour le changement de la pièce la plus importante pour le fonctionnement du Turtle Bot 3 Burger à savoir le capteur LDS-01 sont optimisées à deux niveaux (Figure 46) :

- premièrement par la diminution de la longueur de la séquence par la minimisation du nombre de pièces de 25 à 10 pièces ;
- deuxièmement par la diminution du temps d'exécution de la séquence pour l'assemblage de 620 secondes à 220 secondes et pour le désassemblage de 600 secondes à 180 secondes ce qui montre l'efficacité de l'approche proposée.

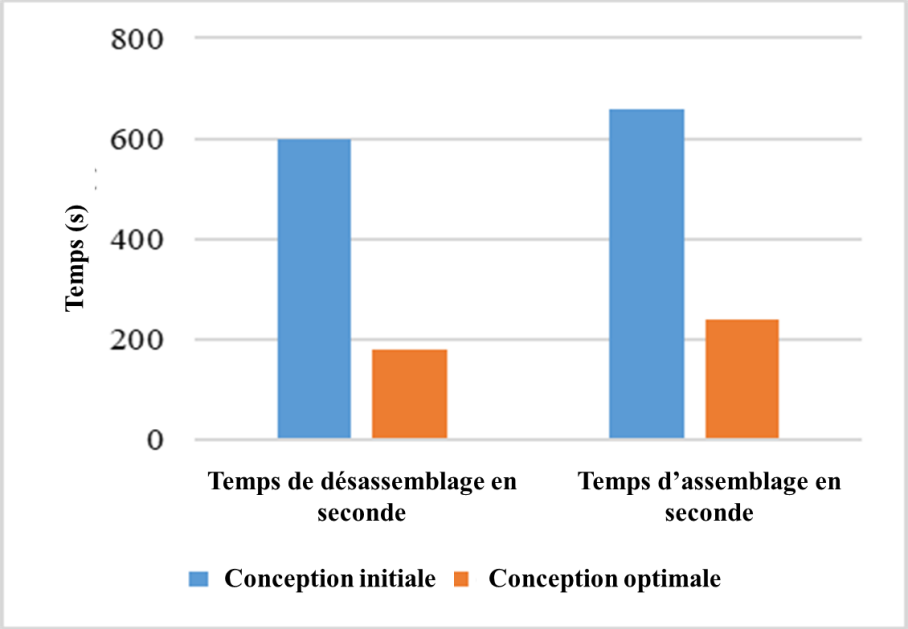


Figure 46 : Temps d'assemblage et de désassemblage (en secondes)

5 CONCLUSION

Ce chapitre a présenté une nouvelle approche d'optimisation des SA/SD partielles et complètes par la reconception de l'architecture CAO des produits. Dans un premier temps, les hypothèses de l'approche ont été introduites, avant de présenter un exemple de produit mécatronique. Ensuite, l'approche proposée a été détaillée et validée sur ce robot TurtleBot3 Burger pour montrer son efficacité. La méthode d'optimisation proposée a permis de réduire l'architecture existante de 25 pièces à 10 pièces dans les SA/SD ainsi qu'un gain de temps de manipulation de plus de la moitié par rapport à la séquence initiale. En effet, le temps d'exécution de l'assemblage est passé de 620 secondes à 220 secondes et pour le désassemblage de 600 secondes à 180 secondes, cela démontre l'efficacité de l'approche proposée.

CONCLUSION GENERALE ET PERSPECTIVES

La génération automatique des plans de désassemblage des mécanismes s'avère une tâche importante pour l'ingénieur dans plusieurs secteurs industriels notamment le secteur de l'automobile et de l'aéronautique. De ce fait, la simulation des plans de désassemblage devient le sujet de plusieurs travaux de recherche. Bien que ces travaux de recherches aient contribué à la génération des séquences de désassemblages (SD), la génération automatique de ces SD à partir d'un modèle CAO reste limitée. De ce fait, les méthodes basées sur l'apprentissage par renforcement semblent être bien adaptées à cette problématique.

L'objectif de cette thèse a été de proposer et d'implémenter une nouvelle approche pour l'optimisation des SD basée sur l'algorithme Q-Network en utilisant la technique d'apprentissage par renforcement. Les approches proposées ont pour objectif le désassemblage d'un mécanisme dans le contexte de la maintenance préventive de pièces d'usure.

La stratégie de recherche adoptée dans ce travail peut se traduire en étapes suivantes :

- Développer une approche d'optimisation pour la génération des SD optimales et faisables en se basant sur le modèle CAO d'un assemblage ;
- Considérer des critères spécifiques pour la maintenance préventive des produits mécaniques afin de faciliter l'accessibilité aux composants d'usure.
- Effectuer une comparaison entre la méthode développée et deux méthodes différentes issues de la littérature.
- Développer une approche pour l'optimisation des SD à travers la reconception des produits existant.
- Evaluer l'approche proposée à travers des exemples de validation.

Ce mémoire de thèse est articulé autour de quatre principaux chapitres. Le premier chapitre de ce mémoire présente un état de l'art sur les différents travaux de recherche liés à notre problématique suivi par une synthèse sur les avantages et les inconvénients des approches proposées. Cela nous a permis de cibler nos objectifs de recherche. Les trois chapitres suivants s'intéressent au développement de nos approches proposées pour la génération des plans optimaux et faisables de désassemblage. Ces approches sont basées essentiellement sur l'exploitation et l'exploration de l'environnement CAO d'un assemblage mécanique. L'algorithme utilisé est un algorithme d'apprentissage par renforcement 'Q-learning' présenté dans le deuxième chapitre. Le troisième chapitre porte sur une étude comparative de l'approche proposée avec deux travaux de littérature. Le dernier chapitre a fait l'objet d'une nouvelle approche d'optimisation des séquences d'assemblage et de désassemblage par la reconception des architecture CAO afin de faciliter l'accès aux pièces d'usure lors de l'assemblage et le désassemblage.

A la lumière de ce travail, les points suivants résument les principaux avantages de l'approche

proposée :

- Minimiser le temps d'exécution pour la génération de séquence de désassemblage
- Minimiser le nombre de changements de direction lors des opérations de démontage
- Minimiser le nombre de changements d'outil lors des opérations de démontage
- Favoriser l'accès aux pièces d'usure lors du démontage pour la maintenance
- Favoriser le démontage des pièces de petite taille en premier lieu
- Optimiser l'architecture CAO afin d'optimiser la séquence de désassemblage pour la maintenance.

L'approche proposée présente également certaines limitations pouvant faire l'objet de travaux futurs à savoir :

- Compléter la séquence optimale générée avec un support visuel qui montre les détails de désassemblage de chaque pièce.
- Intégrer plus de paramètres industriels pour l'optimisation des séquences de désassemblage tels que : l'équilibrage des lignes de désassemblage, l'espace de travail etc.
- Optimiser les séquences de démontage pour la collaboration robot humain.

RÉFÉRENCES

Adel, A., 2022. Future of industry 5.0 in society: human-centric solutions, challenges and prospective research areas. J Cloud Comp 11, 40. https://doi.org/10.1186/s13677-022-00314-5

Aicha, M., Belhadj, I., Hammadi, M., Aifaoui, N., 2022. A mathematical formulation for processing time computing in disassembly lines and its optimization. Computers & Industrial Engineering 165, 107933. https://doi.org/10.1016/j.cie.2022.107933

Aicha, M., Belhadj, I., Hammadi, M., Aifaoui, N., 2021. A Coupled Method for Disassembly Plans Evaluation Based on Operating Time and Quality Indexes Computing. Int. J. of Precis. Eng. and Manuf.-Green Tech. https://doi.org/10.1007/s40684-021-00393-w

Alfadhlani, Samadhi, T.M.A.A., Ma'Ruf, A., Toha, I.S., 2011. AUTOMATIC COLLISION DETECTION FOR ASSEMBLY SEQUENCE PLANNING USING A THREE-DIMENSIONAL SOLID MODEL. J. Adv. Manuf. Syst. 10, 277–291. https://doi.org/10.1142/S021968671100220X

Belhadj, I., Aicha, M., Aifaoui, N., 2022. Product disassembly planning and task allocation based on human and robot collaboration. Int J Interact Des Manuf 16, 803–819. https://doi.org/10.1007/s12008-022-00908-y

Belhadj, I., Trigui, M., Benamara, A., 2017. subassembly identification method based on CAD data, in: Eynard, B., Nigrelli, V., Oliveri, S.M., Peris-Fajarnes, G., Rizzuti, S. (Eds.), Advances on Mechanics, Design Engineering and Manufacturing, Lecture Notes in Mechanical Engineering. Springer International Publishing, Cham, pp. 55–62. https://doi.org/10.1007/978-3-319-45781-9_6

Cai, Y.-J., Choi, T.-M., 2019. Extended producer responsibility: A systematic review and innovative proposals for improving sustainability. IEEE transactions on engineering management 68, 272–288.

Chen, S., Fang, S., Tang, R., 2019. A reinforcement learning based approach for multi-projects scheduling in cloud manufacturing. International Journal of Production Research 57, 3080–3098. https://doi.org/10.1080/00207543.2018.1535205

Chen, S.-F., Liu, Y.-J., 2001. An adaptive genetic assembly-sequence planner. International Journal of Computer Integrated Manufacturing 14, 489–500.

Chung, C., Peng, Q., 2005. An integrated approach to selective-disassembly sequence planning. Robotics and Computer-Integrated Manufacturing 21, 475–485. https://doi.org/10.1016/j.rcim.2004.11.008

Demoly, F., n.d. Conception intégrée et gestion d'informations techniques: application à l'ingénierie du produit et de sa séquence d'assemblage.

Dini, G., Santochi, M., 1992. Automated sequencing and subassembly detection in assembly planning. CIRP annals 41, 1–4.

Dong, T., Zhang, L., Tong, R., Dong, J., 2006. A hierarchical approach to disassembly sequence planning for mechanical product. Int J Adv Manuf Technol 30, 507–520. https://doi.org/10.1007/s00170-005-0036-7

Dorigo, M., Gambardella, L.M., 1997. Ant colony system: a cooperative learning approach to the traveling salesman problem. IEEE Trans. Evol. Computat. 1, 53–66. https://doi.org/10.1109/4235.585892

ElSayed, A., Kongar, E., Gupta, S.M., Sobh, T., 2012. A Robotic-Driven Disassembly Sequence Generator for End-Of-Life Electronic Products. J Intell Robot Syst 68, 43–52. https://doi.org/10.1007/s10846-012-9667-8

Failli, F., Dini, G., 2001. Optimization of disassembly sequences for recycling of end-of-life products by using a colony of ant-like agents. Presented at the Engineering of Intelligent Systems: 14th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, IEA/AIE 2001 Budapest, Hungary, June 4–7, 2001 Proceedings 14, Springer, pp. 632–639.

Fan, S.-K.S., Fan, C., Yang, J.-H., Liu, K.F.-R., 2013. Disassembly and recycling cost analysis of waste notebook and the efficiency improvement by re-design process. Journal of cleaner production 39, 209–219.

Ghandi, S., Masehian, E., 2015. Review and taxonomies of assembly and disassembly path planning problems and approaches. Computer-Aided Design 67–68, 58–86. https://doi.org/10.1016/j.cad.2015.05.001

Gungor, A., Gupta, S.M., 1998a. Disassembly sequence planning for products with defective parts in product recovery. Computers & Industrial Engineering 35, 161–164. https://doi.org/10.1016/S0360-8352(98)00047-3

Gungor, A., Gupta, S.M., 1998b. Disassembly sequence planning for products with defective parts in product recovery. Computers & Industrial Engineering 35, 161–164. https://doi.org/10.1016/S0360-8352(98)00047-3

Guo, X., Zhou, M., Abusorrah, A., Alsokhiry, F., Sedraoui, K., 2021. Disassembly Sequence Planning: A Survey. IEEE/CAA J. Autom. Sinica 8, 1308–1324. https://doi.org/10.1109/JAS.2020.1003515

Hadj, R.B., Belhadj, I., Gouta, C., Trigui, M., Aifaoui, N., Hammadi, M., 2018a. An interoperability process between CAD system and CAE applications based on CAD data. Int J Interact Des Manuf 12, 1039–1058. https://doi.org/10.1007/s12008-017-0445-5

Hadj, R.B., Belhadj, I., Gouta, C., Trigui, M., Aifaoui, N., Hammadi, M., 2018b. An interoperability process between CAD system and CAE applications based on CAD data. Int J Interact Des Manuf 12, 1039–1058. https://doi.org/10.1007/s12008-017-0445-5

Hao, W., n.d. Using genetic/simulated annealing algorithm to solve disassembly sequence planning 7.

Henderson, K., Salado, A., 2021. Value and benefits of model-based systems engineering (MBSE): Evidence from the literature. Systems Engineering 24, 51–66.

Herrmann, J.W., Cooper, J., Gupta, S.K., Hayes, C.C., Ishii, K., Kazmer, D., Sandborn, P.A., Wood, W.H., 2004. New Directions in Design for Manufacturing, in: Volume 3d: 8th Design for Manufacturing Conference. Presented at the ASME 2004 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, ASMEDC, Salt Lake City, Utah, USA, pp. 853–861. https://doi.org/10.1115/DETC2004-

57770
Homem De Mello, L.S., Sanderson, A.C., 1988. Automatic Generation of Mechanical Assembly Sequences: Defense Technical Information Center, Fort Belvoir, VA. https://doi.org/10.21236/ADA204234
Huang, W., Xu, Q., 2017. Automatic generation and optimization of stable assembly sequence based on ACO algorithm, in: 2017 IEEE International Conference on Mechatronics and Automation (ICMA). Presented at the 2017 IEEE International Conference on Mechatronics and Automation (ICMA), IEEE, Takamatsu, Japan, pp. 2057–2062. https://doi.org/10.1109/ICMA.2017.8016135
Huang, Y.M., Huang, C.-T., 2002. Disassembly matrix for disassembly processes of products. International Journal of Production Research 40, 255–273. https://doi.org/10.1080/00207540110079770
Issaoui, L., Aifaoui, N., Benamara, A., 2015. Solution space reduction of disassembly sequences generated automatically via computer aids. Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science 229, 2977–2986. https://doi.org/10.1177/0954406214565803
Kamble, S.S., Gunasekaran, A., Ghadge, A., Raut, R., 2020. A performance measurement system for industry 4.0 enabled smart manufacturing system in SMMEs-A review and empirical investigation. International journal of production economics 229, 107853.
Kheder, M., Trigui, M., Aifaoui, N., 2017. Optimization of disassembly sequence planning for preventive maintenance. The International Journal of Advanced Manufacturing Technology 90, 1337–1349.
Kheder, M., Trigui, M., Aifaoui, N., 2015a. Disassembly sequence planning based on a genetic algorithm. Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science 229, 2281–2290. https://doi.org/10.1177/0954406214557340
Kheder, M., Trigui, M., Aifaoui, N., 2015b. Disassembly sequence planning based on a genetic algorithm. Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science 229, 2281–2290. https://doi.org/10.1177/0954406214557340
Kwiatkowski, A., Alvarado, E., Kalogeiton, V., Liu, C.K., Pettré, J., van de Panne, M., Cani, M., 2022. A survey on reinforcement learning methods in character animation. Presented at the Computer Graphics Forum, Wiley Online Library, pp. 613–639.
Li, J.R., Khoo, L.P., Tor, S.B., 2005. An object-oriented intelligent disassembly sequence planner for maintenance. Computers in Industry 56, 699–718. https://doi.org/10.1016/j.compind.2005.03.005
Liu, J., Zhou, Z., Pham, D.T., Xu, W., Ji, C., Liu, Q., 2018. Robotic disassembly sequence planning using enhanced discrete bees algorithm in remanufacturing. International Journal of Production Research 56, 3134–3151. https://doi.org/10.1080/00207543.2017.1412527
Liu, Q., Liu, Z., Xu, W., Tang, Q., Zhou, Z., Pham, D.T., 2019. Human-robot collaboration in disassembly for sustainable manufacturing. International Journal of Production Research 57,

4027–4044. https://doi.org/10.1080/00207543.2019.1578906
Liu, Xinhua, Peng, G., Liu, Xiumei, Hou, Y., 2012. Disassembly sequence planning approach for product virtual maintenance based on improved max–min ant system. Int J Adv Manuf Technol 59, 829–839. https://doi.org/10.1007/s00170-011-3531-z
Liu, Z., Liu, Q., Wang, L., Xu, W., Zhou, Z., 2021. Task-level decision-making for dynamic and stochastic human-robot collaboration based on dual agents deep reinforcement learning. Int J Adv Manuf Technol 115, 3533–3552. https://doi.org/10.1007/s00170-021-07265-2
Luo, Y., Peng, Q., Gu, P., 2016. Integrated multi-layer representation and ant colony search for product selective disassembly planning. Computers in Industry 75, 13–26.
Malik, S., 2010. Performance comparison between ant algorithm and modified ant algorithm. International Journal of Advanced Computer Science and Applications 1.
Mao Huang, Y., Liao, Y., 2009. Disassembly processes with disassembly matrices and effects of operations. Assembly Automation 29, 348–357.
Marian, R.M., Luong, L.H., Abhary, K., 2006. A genetic algorithm for the optimisation of assembly sequences. Computers & Industrial Engineering 50, 503–527.
Mathew, A., Rao, C.S.P., 2010. A CAD system for extraction of mating features in an assembly. Assembly Automation 30, 142–146. https://doi.org/10.1108/01445151011029772
May, R.M., Goebbert, K.H., Thielen, J.E., Leeman, J.R., Camron, M.D., Bruick, Z., Bruning, E.C., Manser, R.P., Arms, S.C., Marsh, P.T., 2022. MetPy: A Meteorological Python Library for Data Analysis and Visualization. Bulletin of the American Meteorological Society 103, E2273–E2284. https://doi.org/10.1175/BAMS-D-21-0125.1
Meng, K., Lou, P., Peng, X., Prybutok, V., 2016. An improved co-evolutionary algorithm for green manufacturing by integration of recovery option selection and disassembly planning for end-of-life products. International Journal of Production Research 54, 5567–5593. https://doi.org/10.1080/00207543.2016.1176263
Özceylan, E., Kalayci, C.B., Güngör, A., Gupta, S.M., 2019. Disassembly line balancing problem: a review of the state of the art and future directions. International Journal of Production Research 57, 4805–4827. https://doi.org/10.1080/00207543.2018.1428775
Pomares, J., Puente, S., Torres, F., Candelas, F., Gil, P., 2004. Virtual disassembly of products based on geometric models. Computers in industry 55, 1–14.
Pornsing, C., Watanasungsuit, A., 2014. Discrete particle swarm optimization for disassembly sequence planning, in: 2014 IEEE International Conference on Management of Innovation and Technology. Presented at the 2014 IEEE International Conference on Management of Innovation and Technology (ICMIT), IEEE, Singapore, Singapore, pp. 480–485. https://doi.org/10.1109/ICMIT.2014.6942474
Santochi M, Dini G (1991) Flexible and process integrated knowledge bases as instruments for modern CAPP systems. In: Proceedings of CIRP Seminars on Manufacturing Systems, Ljubljana, vol 20, no 3/4, n.d.
Smith, S., Hung, P.-Y., 2015. A novel selective parallel disassembly planning method for green design. Journal of engineering design 26, 283–301.
Tang, Y., 2002. Disassembly Modeling, Planning, and Application. Journal of Manufacturing

Systems.
Torres, F., Puente, S.T., Aracil, R., 2003. Disassembly Planning Based on Precedence Relations among Assemblies. The International Journal of Advanced Manufacturing Technology 21, 317–327. https://doi.org/10.1007/s001700300037
Tseng, H.-E., Chang, C.-C., Lee, S.-C., Huang, Y.-M., 2018. A Block-based genetic algorithm for disassembly sequence planning. Expert Systems with Applications 96, 492–505. https://doi.org/10.1016/j.eswa.2017.11.004
Tseng, Y.-J., Kao, H.-T., Huang, F.-Y., 2010. Integrated assembly and disassembly sequence planning using a GA approach. International Journal of Production Research 48, 5991–6013. https://doi.org/10.1080/00207540903229173
Tuncel, E., Zeid, A., Kamarthi, S., 2014. Solving large scale disassembly line balancing problem with uncertainty using reinforcement learning. J Intell Manuf 25, 647–659. https://doi.org/10.1007/s10845-012-0711-0
Vyas, P., Rickli, J.L., 2016. Automatic Extraction and Synthesis of Disassembly Information From CAD Assembly STEP File, in: Volume 4: 21st Design for Manufacturing and the Life Cycle Conference; 10th International Conference on Micro- and Nanosystems. Presented at the ASME 2016 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, American Society of Mechanical Engineers, Charlotte, North Carolina, USA, p. V004T05A042. https://doi.org/10.1115/DETC2016-59577
Wang, H., Peng, Q., Zhang, J., Gu, P., 2017. Selective Disassembly Planning for the End-of-life Product. Procedia CIRP 60, 512–517. https://doi.org/10.1016/j.procir.2017.02.003
Wang, J., Liu, J., Li, S., Zhong, Y., 2003. Intelligent selective disassembly using the ant colony algorithm. Ai Edam 17, 325–333.
Wang, J.F., Liu, J.H., Zhong, Y.F., 2005. A novel ant colony algorithm for assembly sequence planning. The international journal of advanced manufacturing technology 25, 1137–1143.
Watanabe, K., Inada, S., 2020. Search algorithm of the assembly sequence of products by using past learning results. International Journal of Production Economics 226, 107615. https://doi.org/10.1016/j.ijpe.2020.107615
Xia, K., Gao, L., Li, W., Wang, L., Chao, K.-M., 2014. A Q-learning based selective disassembly planning service in the cloud based remanufacturing system for WEEE. Presented at the International Manufacturing Science and Engineering Conference, American Society of Mechanical Engineers, p. V001T04A012.
Yu, J., Xu, L.D., Bi, Z., Wang, C., 2014. Extended Interference Matrices for Exploded View of Assembly Planning. IEEE Trans. Automat. Sci. Eng. 11, 279–286. https://doi.org/10.1109/TASE.2012.2235144
Zhang Gang, Deng Kewen, Hou Qiang, Lai Shengjing, 2006. Research on assembly modeling and sequence planning based on features, in: International Technology and Innovation Conference 2006 (ITIC 2006). Presented at the International Technology and Innovation Conference 2006 (ITIC 2006), IEE, Hangzhou, China, pp. 408–411. https://doi.org/10.1049/cp:20060795
Zhang, H., Liu, P., Guo, X., Wang, J., Qin, S., Qi, L., Zhao, J., 2022. An Improved Q-Learning

Algorithm for Solving Disassembly Line Balancing Problem Considering Carbon Emission, in: 2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC). Presented at the 2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC), IEEE, Prague, Czech Republic, pp. 872–877. https://doi.org/10.1109/SMC53654.2022.9945321

Zhang, H., Peng, Q., Zhang, J., Gu, P., 2021. Planning for automatic product assembly using reinforcement learning. Computers in Industry 130, 103471. https://doi.org/10.1016/j.compind.2021.103471

Zhang, H.C., Kuo, T.C., n.d. ased Disassembly Sequence Planning for EOL Product Recycling 12.

Zhang, X.F., Zhang, S.Y., 2010. Product cooperative disassembly sequence planning based on branch-and-bound algorithm. Int J Adv Manuf Technol 51, 1139–1147. https://doi.org/10.1007/s00170-010-2682-7

Zhao, X., Li, C., Tang, Y., Cui, J., 2021. Reinforcement Learning-Based Selective Disassembly Sequence Planning for the End-of-Life Products With Structure Uncertainty. IEEE Robot. Autom. Lett. 6, 7807–7814. https://doi.org/10.1109/LRA.2021.3098248

Zhong, L., Youchao, S., Ekene Gabriel, O., Haiqiao, W., 2011. Disassembly sequence planning for maintenance based on metaheuristic method. Aircraft Engineering and Aerospace Technology 83, 138–145. https://doi.org/10.1108/00022661111131221

Zhou, M., DiCesare, F., 2012. Petri net synthesis for discrete event control of manufacturing systems. Springer Science & Business Media.

Zhou, M., Venkatesh, K., 1999. Modeling, simulation, and control of flexible manufacturing systems: a Petri net approach. World Scientific.

Zhou, Z., Liu, J., Pham, D.T., Xu, W., Ramirez, F.J., Ji, C., Liu, Q., 2019. Disassembly sequence planning: Recent developments and future trends. Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture 233, 1450–1471. https://doi.org/10.1177/0954405418789975

Zhu, B., Sarigecili, M.I., Roy, U., 2013. Disassembly information model incorporating dynamic capabilities for disassembly sequence generation. Robotics and Computer-Integrated Manufacturing 29, 396–409.

ANNEXES :

ANNEXE 1 : MODELISATION MATRICIELLE

LES MATRICES DE CONTACT :

Les matrices de contact entre les composants d'un assemblage sont déterminées en détectant les contacts le long des trois axes X, Y et Z. Les éléments de chaque ligne de la matrice représentent les composants qui sont en contact avec le composant étudié après assemblage suivant une direction donnée. Si un élément de la matrice est égal à 1, cela signifie qu'il y a un contact entre ces deux composants le long de la direction correspondante. Sinon, l'élément est égal à 0 (Mao Huang and Liao, 2009). Les matrices de contact, selon les directions X et Y et Z, d'un exemple contenant trois composants sont présentées sur la Figure 47 :

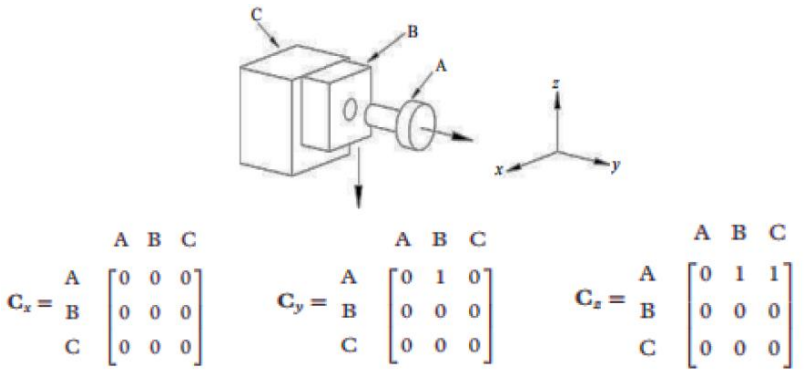


Figure 47: Matrices de contact selon les trois direction (X, Y, Z) (Mao Huang and Liao, 2009)

LES MATRICES DE CONNEXION :

La modélisation d'un produit mécanique à l'aide de matrices de connexion fonctionne selon un principe similaire à celui des matrices de contact. Cependant, les matrices de connexion fournissent en plus des informations sur le type de contact en attribuant des valeurs prédéfinies aux éléments de la matrice (Figure 48). Un exemple de scores a été défini par (Dini and Santochi, 1992) et présenté sur la Figure 49.

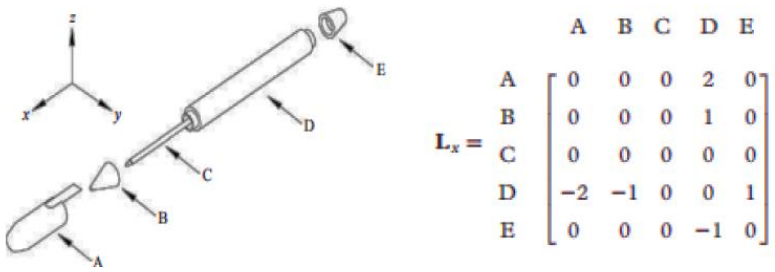


Figure 48: Exemple de matrice de connexion (Huang and Huang, 2002)

$C_{i,j}$	CONNECTION BETWEEN e_i AND e_j	DISASSEMBLABILITY OF e_i
1	THREADED CONNECTION	YES
-1	THREADED CONNECTION	NO
2	DRIVE FIT	YES
-2	DRIVE FIT	NO
3	ELASTIC RING	YES
-3	ELASTIC RING	NO
4	O-RING	YES
-4	O-RING	NO
0	ABSENCE OF THE PREVIOUS CONNECTIONS	YES

Figure 49: Exemple de scores de matrice de connexion définis par (Dini and Santochi, 1992)

Ces matrices peuvent être construites de deux manières : soit manuellement, mais le processus est très chronophage, surtout en cas de changement de pièce ; soit elles peuvent être extraites automatiquement à partir de la CAO, en s'appuyant sur des algorithmes qui facilitent leur exploitation (Mathew and Rao, 2010, Alfadhlani et al., 2011, Hadj et al., 2018, Vyas and Rickli, 2016, Belhadj et al., 2017). Ainsi, en se basant sur des données CAO et des méthodes de représentation par graphe, de nombreuses approches PSD avec différents objectifs, outils et modes (destructif ou non, partiel/total) ont été développées.

LES MATRICES D'ADJACENCE :

La matrice d'adjacence est apparue avec la méthode de F. Torres et al. qui ont utilisé un algorithme pour déterminer SD optimale d'un produit en se basant sur des graphes d'adjacence (Torres et al., 2003). Ensuite, F. Demoly a utilisé ces graphes d'adjacence pour les transformer en matrices d'adjacence et les exploiter dans son étude de conception intégrée et de gestion de connaissances techniques appliqué à la planification des séquences d'assemblage (Demoly, 2010). Cette matrice permet de décrire les précédences entre des pièces sans contact direct (Figure 50). La matrice d'adjacence [Pn] s'exprime comme suit : c'est une matrice carrée et symétrique, dont la taille est égale à (n×n), où n représente le nombre total de pièces. L'élément P(i,j) représente un contact existant entre deux pièces, i et j, et peut avoir trois attributs possibles comme suit :

- 1 : il y a un contact direct entre Pi et Pj
- 0 : il n'y a pas un contact entre Pi et Pj
- λ : il n'y a pas un contact direct entre Pi et Pj, mais Pi doit être assemblé avant(λ)/après(-λ) Pj.

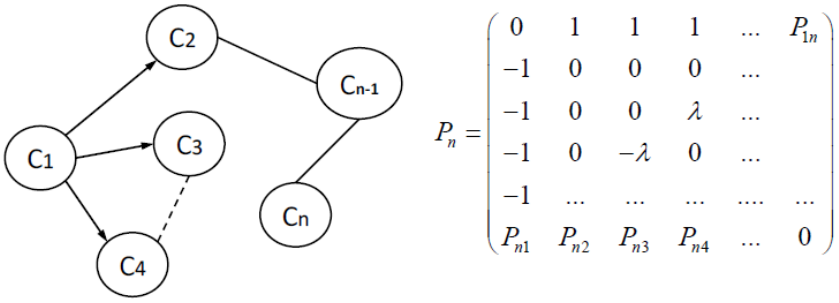


Figure 50: Graphe et matrice d'adjacence (Demoly, 2010)

LES MATRICES D'INTERFERENCES :

Les matrices d'interférences permettent de détecter les collisions (contact ponctuel lors d'un mouvement) entre les pièces d'un assemblage, Elles sont souvent générées avec un test de collision dynamique (manuel ou par simulation), en déplaçant le composant étudié dans les trois directions (X, Y, Z). En général, il y a trois matrices, une pour chaque axe (Figure 51). Ces matrices sont carrées et symétriques, avec une taille de (n×n), où n représente le nombre total de pièces présentes dans l'assemblage mécanique. Chaque élément Nij d'une matrice est déterminé selon la formule suivante :

- Nij = 1 s'il existe une collision entre les pièces i et j ;
- Nij = 0, sinon.

Certains auteurs utilisent ces matrices pour générer des SD (Huang and Huang, 2002).

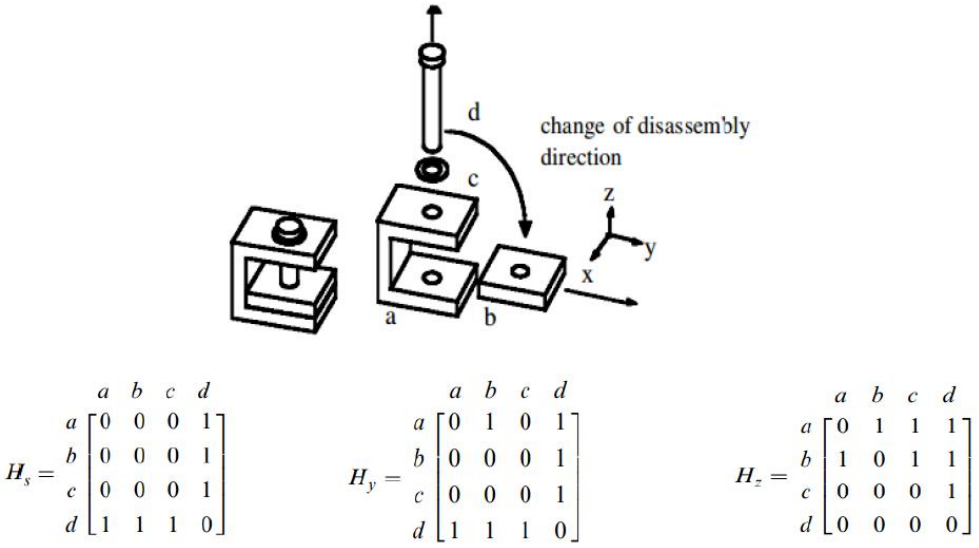


Figure 51: Exemples de matrices d'interférence selon les trois axes (X, Y, Z) (Huang and Huang, 2002)

ANNEXE 2 : LES PRINCIPES FONDAMENTAUX DES ALGORITHMES DE Q-LEARNING

LES FONDAMENTAUX DU QN

QN est au cœur des algorithmes d'apprentissage par Q-learning. Il s'agit d'un algorithme utilisé dans la méthode d'apprentissage par renforcement qui permet à un agent d'apprendre à prendre des décisions optimales dans un environnement donné. L'agent exploite l'environnement et apprend à prendre des décisions optimales en effectuant des actions et en recevant des récompenses. QN est un algorithme basé sur des valeurs et est donc adapté aux environnements discrets. La propriété la plus importante d'un réseau Q est son temps d'exécution rapide. Ce processus d'apprentissage est basé sur plusieurs paramètres tels que :

- La fonction Q (S, a) représente la valeur pour l'état S et l'action a. La valeur Q exprime la qualité de l'action effectuée en fonction de la récompense R. Le Q-learning utilise souvent une "table Q" pour représenter les valeurs discrètes de la fonction Q. Cette table a une ligne pour chaque état possible de l'environnement et une colonne pour chaque action possible. Ainsi, chaque cellule de la table Q représente la valeur Q d'une paire état-action. Par exemple, supposons que nous ayons un environnement avec 3 états possibles (S1, S2, S3) et 2 actions possibles (a1, a2). La table Q serait alors la suivante :

Tableau 29: Exemple de Q-table

State/action	a1	a2
S1	Q(1,1)	Q(1,2)
S2	Q(2,1)	Q(2,2)
S3	Q(3,1)	Q(3,2)

Le réseau Q est un algorithme itératif qui explore progressivement l'espace des états et des actions, et ajuste les valeurs de la fonction Q pour converger vers les valeurs optimales.

La description de cet algorithme est donnée ci-dessous (Algorithme 1) :

- Le ***taux d'apprentissage α***, qui est compris entre 0 et 1 : $\alpha = 0$ signifie que l'agent n'apprend rien, $\alpha = 1$ signifie que l'agent ignore tout ce qu'il a appris précédemment et ne tient compte que des résultats les plus récents.
- Le ***facteur d'actualisation γ***, qui est compris entre 0 et 1, détermine l'importance des récompenses futures : $\gamma = 0$ rendrait l'agent myope ou à courte vue et $\gamma = 1$ permettrait de prendre en compte

les récompenses plus éloignées, mais risquerait une divergence de Q (S, a).

- La politique ε-greedy (ε), qui est la méthode ou la stratégie de prise de décision de l'agent. Elle détermine comment l'agent sélectionne ses actions en fonction de l'état actuel.

Algorithme 1: Q-network

Entrée : taux d'apprentissage $\alpha > 0$ et un petit $\varepsilon > 0$, taux $\gamma > 0$

Sortie : tableau Q [.,.]

Initialiser Q [s, a] pour tout état s non final, toute action a de façon arbitraire, et Q (état terminal, a) = 0 pour toute action a

Répéter

//début d'un épisode

s : = état initial

Répéter

//étape d'un épisode

Choisir une action a depuis s en utilisant la politique spécifiée par Q (par exemple ε-greedy)

Exécuter l'action a

Observer la récompense r et le nouvel état s'

$Q[s, a] := \alpha [r + \gamma \max_{a'} Q(s', a')]$

s := s'

jusqu'à ce que s soit l'état terminal

LES FONDAMENTAUX DE DQN

Le cœur d'un algorithme DQN est un réseau Q, L'avantage de DQN est sa capacité d'exécution sur des environnements continus. Les algorithmes DQN sont adaptés à la résolution de problèmes dynamiques et peuvent générer des animations 3D, comme dans les jeux vidéo ou différentes simulations industrielles. Cependant, leur espace d'état et leur taille peuvent être extrêmement grand, ce qui peut entraîner des temps d'exécution longs (Sewak, 2019). Le DQN dispose d'une mémoire de

lecture dans laquelle il enregistre l'historique de ses expériences précédentes (valeurs Q) pour les comparer aux nouvelles prédictions. Ainsi, l'identification de la valeur Q repose sur les processus d'exploration et d'exploitation.

Lors de l'entraînement du DQN, l'exploration est initiée et l'agent commence ses premières étapes d'entraînement avec une mémoire de répétition vide. L'agent explore la matrice de récompense pour calculer la Q-cible et atteindre son état objectif. Pendant l'étape d'exploration, l'agent DQN enregistre la Q-cible déjà calculée. Une fois que la mémoire de répétition est remplie, l'agent DQN entame le processus d'exploitation, où il exploite la mémoire de répétition pour comparer les valeurs Q prédites sauvegardées dans la mémoire de répétition avec les nouvelles valeurs de Q-cible calculées, en utilisant une fonction de perte de moindres carrés (MSE) (Figure 52).

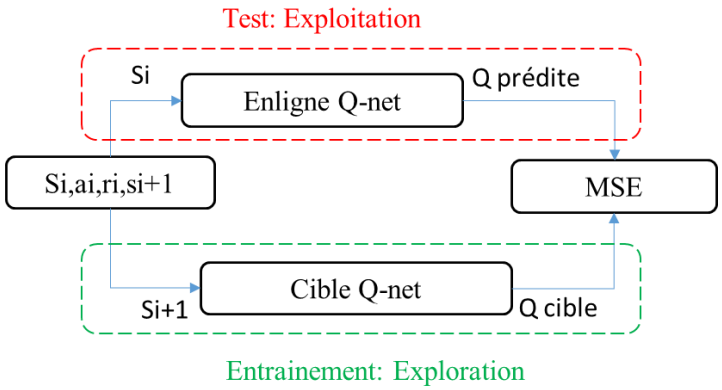


Figure 52: Le schéma d'exploitation et d'exploration du DQN

Dans le pseudocode du DQN, décrit sur l'algorithme 2 :

- D représente la mémoire de répétition.
- Q avec des poids aléatoires θ représente le réseau Q en ligne.
- Qˆ avec des poids $\boldsymbol{\theta}^{-} = \boldsymbol{\theta}$ représente le réseau Q cible.
- Dans la politique epsilon-greedy, sélectionner at = argmaxa Q(ϕ(st), a; θ).
- Définir yj comme la cible de différence temporelle (TD), qui est le calcul de la Q-cible en utilisant le réseau Q cible.
- Effectuer une étape de descente de gradient sur $(yj - Q(\phi j, aj' ; \theta^{-}))^2$ pour calculer la perte et mettre à jour le réseau Q, où l'algorithme effectue le calcul de la Q prédite en utilisant le réseau en ligne et le calcul de la perte entre la Q-cible et la Q-prédite.
- Toutes les C étapes, réinitialiser Qˆ = Q : mise à jour du réseau Q cible.

Algorithme 2: Deep Q-network

Initialiser la mémoire de relecture D à la capacité N

Initialiser la fonction action-valeur Q avec des poids aléatoires θ

Initialiser la fonction de valeur d'action cible Q^ avec le poids $\theta^- = \theta$

Pour l'épisode 1..M faire

Initialiser la séquence S 1 = {x1} et la séquence prétraitée ϕ1 = ϕ(s1)

Pour t = 1..T faire

Avec une probabilité ε, choisir une action aléatoire a_t;

Sinon sélectionner $a_t = \text{argmax}_a Q(\phi(s_t), a; \theta)$

Exécuter l'action at dans l'émulateur et observer la récompense rt

$s_{t+1} = s_t$, at, xt+1, ϕt+1 = ϕ(st+1)

Enregistrer l'expérience (ϕ_t, a_t, r_t, ϕ_{t+1}) dans D

Échantillonner un mini-ensemble aléatoire d'expériences :

(ϕt, a_t, r_t, ϕ_{t+1}) de D

$$y_j = \begin{cases} r_j & \text{si l'épisode se termine à l'étape } j+1 \\ r_j + \gamma \max_{a'} \hat{Q}(\phi_{j+1}, a'; \theta^-) & \end{cases}$$

Effectuer une descente de gradient sur $(y_j - Q(\phi_j, a_j; \theta^-))^2$

Pour le poids θ

A chaque étape C réinitialiser Q^ = Q

Fin pour

ANNEXE 3 : SYNTHESE DE L'ETAT DE L'ART

Tableau 30 : Analyse de l'état de l'art

Référence/Auteurs	Modélisation / Représentation du produit assemblé	Optimisation du PSD		Avantages	Inconvénients
		Méthode	Paramètres		
(Guo et al., 2020)	Etat de l'art de PSD				
(Özceylan et al., 2019), (Liu et al., 2018), (Ghandi and Masehian, 2015), (Meng et al., 2016), (Smith and Hung, 2015), (Marconi et al., 2019)	-	-	-	Classification des types de démontage	-
(Tang, 2002)		Etat de l'art de PSD			
(Homem De Mello and Sanderson, 1988)	Méthode des graphes	-	La représentation des contraintes mécaniques	Tester la faisabilité des opérations d'assemblage	Génération des séquences de montage sans optimisation
(Dini and Santochi, 1992), (Zhang and Kuo, 1997)		-	-	Génération du SD	Génération manuelle du PSD
(Chung and Peng, 2005)		-	-	Extraction des PSDs des plateformes CAO	Pas d'optimisation de PSD
(Dong et al., 2006)		Graphe de liaison attribué	-	Génération du DSP	Pas d'optimisation de PSD
(Zhang and Kuo, 1997)				Génération du DSP	Génération manuelle du DSP
(Zhang and Zhang, 2010)		L'algorithme branch-and-bound	-	Générer un arbre hiérarchique de désassemblage coopératif	Pas d'optimisation de PSD
(Wang et al., 2017)		-	La représentation des contraintes mécanique	Identifier les démontages destructifs en non destructifs	Pas de génération de SD
(Issaoui et al., 2015)		-	Identification de la SD par la méthode d'élimination	Génération automatisée de séquences de désassemblage	Pas d'optimisation ciblée
(Li et al., 2005)		Graphes de contraintes	Démontage pour la maintenance	Proposition de prototype de planificateur de séquence de démontage de maintenance	Utilisation difficile avec un nombre de composants élevé
(Zhou and DiCesare, 2012), (Zhou and Venkatesh, 1999)	Réseaux de Petri Net (PN)		Coût et avantages environnementaux de processus de désassemblage	Génération de séquence de désassemblage	
(Kuo and Wang, 2010), (Giudice, 2010),(Kuo, 2013), (Guo et al., 2015)	Réseaux de Petri Net (PN)	-	PSD de système mécatronique		
(ElSayed et al., 2012), (Zhang Gang et al., 2006), (Yu et al., 2014), centraux (Pornsing and Watanasungsuit,2014), (Zhu et al., 2013), (Luo et al., 2016), (Pomares et al., 2004)	Matrices	Etat de l'art de PSD avec la représentation matricielle des contraintes entre les composants			
(Dini and Santochi, 1992)				Définition des scores de matrice de connexion	
(Santochi M, Dini G 1991)		-	-	Génération de séquence de désassemblage	Extraction manuelle des matrices
(Gungor and Gupta, 1998)		-	Recyclage et récupération des pièces		-
(Li et al., 2002) (Hao, 2008) (Tseng et al, 2010)		-	Démontage pour la maintenance et la fin de vie		-
		Graphe et matrice d'adjacence	-		
(Huang and Huang, 2002)		Matrice d'interférence	Direction de démontage		Un seul paramètre d'optimisation
(Mao Huang and Liao, 2009)		Matrice de contact	-		-
Huang and Huang, 2002)		Matrice de connexion	-		-
(Mathew and Rao, 2010, Alfadhlani et al., 2011, Hadj et al., 2018, Vyas and Rickli, 2016, Belhadj et al., 2017)		Exploitation de différentes matrices	-		
(Tseng et al., 2010)		Algorithme Flatworm (FA)	Changement d'outil et de direction	Minimisant le nombre d'outils et de changements de direction dans la SD	Deux paramètres d'optimisation
(Kongar and Gupta, 2006)	-	GA			
(Tseng et al., 2018)		GA basé sur les blocs			
(Dorigo and Gambardella, 1997)		ACO			
(Liu et al., 2012)		Le système de fourmis Max-			
(Kheder et al., 2015)		GA			
(Kheder et al., 2017)	Matrices d'interférence	ACO			
(Luo et al., 2016)					
(Bedeoui et al., 2021)	-	-	Outil de démontage	Simulation de l'opération de démontage avec les outils de démontage	Long temps d'exécution
(Aicha et al., 2021), (Aicha et al., 2022)	-	-	Environnement de travail et temps de démontage	Organisation de l'environnement industriel pour minimiser le temps de démontage	
(Belhadj et al., 2022)	-	-	Collaboration robot humain	Répartition des tâches dans un démontage collaboratif robot-humain	
(Watanabe and Inada, 2020)	Graphe	Apprentissage par renforcement (RL)	Réalisation d'une SD optimale	Utilisation d'un robot pour la réalisation d'une SD optimale	Pas d'optimisation ou génération de SD
(Torres et al., 2003) (Demoly, 2010)				Génération des séquences d'assemblage	
(Zhao et al., 2021)	-	MDP	PSD sélectif	Génération de séquence de désassemblage sélectif	-
(Zhao et al., 2021)	-	RL	Génération de séquence de montage	Algorithme stable et exécution rapide	Pas de prise en compte des outils de montage
(Huang and Xu, 2017)	Graphe et matrice	ACO	Trajectoire de montage et de démontage	La faisabilité géométrique et la stabilité de l'opération d'assemblage/désassemblage	Sensibilité du l'algorithme au réglage des paramètres