Développement d'une méthodologie d'ordonnancement/optimisation adaptée aux systèmes industriels de type HVLV (High-Variety, Low-Volume).

Imed Nasri

UNIVERSITÉ DE GRENOBLE

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ DE GRENOBLE

Spécialité : **Sciences Pour l'Ingénieur**

Arrêté ministériel : 7 août 2006

Présentée par

Imed NASRI

Thèse dirigée par **Georges HABCHI**
Co-dirigée par **Reda BOUKEZZOULA**

Préparée au sein du laboratoire **SYMME**
Dans **l'École Doctorale SISEO**

Développement d'une méthodologie d'ordonnancement/optimisation adaptée aux systèmes industriels de type HVLV (High-Variety, Low-Volume)

Thèse soutenue publiquement le **9 avril 2013**,
devant le jury composé de :

M. Yannick FREIN
Professeur des Universités à l'INP de Grenoble, Président
M. Jean-Louis BOIMOND
Professeur des Universités à l'Université d'Angers, Rapporteur
M. Alexandre DOLGUI
Professeur des Mines à l'Ecole des Mines de Saint-Etienne, Rapporteur
M. Claude MARTINEZ
Maître de Conférences à l'Université de Nantes, Examinateur
M. Georges HABCHI
Professeur des Universités à l'Université de Savoie, Directeur de thèse
M. Reda BOUKEZZOULA
Maître de Conférences à l'Université de Savoie, Co-directeur de thèse

Université Joseph Fourier / Université Pierre Mendès France /
Université Stendhal / Université de Savoie / Grenoble INP

À ma mère Hayet HIDRI et mon père Mohammed-El-Fahem NASRI : c'est du profond de mon cœur que je vous adresse ce mémoire, une mention très spéciale pour vous, car sans votre support rien n'aurait été possible. Je vous remercie pour toutes les prières à mon attention, pour m'avoir fortement soutenu durant ces trois années et de toujours m'avoir remonté le moral surtout pendant la période de rédaction.

De plus, je suis très reconnaissant des forts encouragements et aides de la part de mes très chères sœurs Afef et Dorra NASRI à qui je dédie aussi ce travail. Je n'oublie pas de dédier ce travail aussi au petit : Yassine ELAYEDDI.

Je remercie également du fond de mon cœur ma très chère tante Fayza HIDRI et son époux Mohammed BENHAMED pour leur soutien.

Remerciements

Je remercie chaleureusement tous les membres de mon jury de thèse :

- M. Yannick FREIN, Professeur des universités à l'INP de Grenoble, pour avoir accepté la présidence de ce jury,

- M. Jean-Louis BOIMOND, Professeur des universités à l'Université d'Angers et,

- M. Alexandre DOLGUI, Professeur des mines à l'Ecole des Mines de Saint-Etienne, pour m'avoir fait l'honneur de rapporter les travaux de cette thèse, ainsi que

- M. Claude MARTINEZ, Maître de conférences à l'Université de Nantes de vouloir examiner mes travaux de thèse et pour toutes les discussions très constructives et extrêmement enrichissantes que nous avons eues ensemble.

Je tiens à exprimer ma profonde gratitude et mes remerciements les plus sincères à M. Georges HABCHI, Professeur des universités à l'Université de Savoie et Reda BOUKEZ-ZOULA, Maître de conférences à l'Université de Savoie, pour avoir dirigé mes travaux de thèse mais avant tout pour m'avoir accordé leur confiance et m'avoir attribué le sujet de cette thèse. Les trois années (et six mois et neuf jours !) que j'ai passées sous leur encadrement furent une expérience très riche tant au niveau scientifique et professionnel, qu'au niveau personnel. Je tiens à remercier encore mes encadrants de thèse, pour leurs compétences et connaissances variées, pour la qualité des échanges toujours constructifs et pour les débats passionnants que nous avons eus.

Je souhaite également remercier particulièrement mes collègues de bureau Weiqun LIU et Naima CHOUKET pour leur soutien moral. Mes chaleureux remerciements vont aussi à tous les doctorant(e)s des laboratoires SYMME et LISTIC en particulier Moustafa EL KASS, Nathalie BAUDET et Ahmed GUERMAZI pour leurs visites dans le bureau. J'associe mes remerciements à tous les membres du laboratoire SYMME qui ont aussi rendu cette période très agréable.

Table des matières

Table des figures

Liste des tableaux

Introduction générale

Dans le domaine de la production industrielle, les tendances actuelles indiquent que les systèmes manufacturiers performants doivent s'adapter rapidement aux fluctuations du marché (demandes aléatoires) et aux perturbations internes (pannes des machines, qualité des produits). Ces exigences industrielles imposées par le marché, la concurrence, la qualité ainsi que la densité et la diversité des produits traités entraînent une complexité sans cesse croissante des systèmes de production (SdPs). Un exemple typique contemporain d'un système de production (SdP) complexe, couramment utilisé, dans l'industrie est celui des systèmes HVLV (*High-Variety, Low-Volume*). Ce sont des systèmes capables de s'adapter à une possible évolution de l'environnement industriel. Ils peuvent présenter une diversité importante des flux de produits avec des séquences variées de production et avec une faible cadence de produits. L'objectif associé à ce type de systèmes est alors d'assurer un traitement le plus varié possible avec un maximum de productivité, au moindre coût et dans les délais.

Un autre type de systèmes de production a été largement étudié dans la littérature. Il s'agit des systèmes de production à haute cadence de production et à faible variété de produits (les industries de papier, de l'agro-alimentaire ou encore des semi-conducteurs). Dans ces industries, sur un horizon de six mois à un an, le flux des produits peut être facilement considéré comme continu [5]. Récemment, la méthodologie de régulation des flux de production développée dans les travaux de thèse de Karim Tamani [1] a permis d'apporter quelques éléments de réponse à la problématique de conception, de modélisation et de pilotage de ce genre de systèmes. La stratégie proposée est basée sur l'approximation de la circulation des flux de produits discrets, dans un système de production, par un modèle dit fluide ou continu. Cette modélisation a permis de transposer des concepts de commande et de supervision issus de l'automatique des systèmes continus pour traiter des problèmes de régulation et le suivi de trajectoire de production (débit de production). Toutefois, la faisabilité de cette approche est restreinte à des processus de fabrication à haute cadence et produisant des quantités élevées. En adoptant la même approche, il est clair qu'un pilotage temps réel au niveau « produit » dans les systèmes HVLV requerra une « discrétisation » du modèle continu des flux ainsi que des actions de commande (continues). A ce niveau, il s'agit parfois de résoudre des problèmes complexes d'ordonnancement en présence d'aléas et incertitudes de natures diverses (pannes machines, connaissance im-

précise des paramètres de production ou de la demande, etc.). Dans un tel contexte, il est important d'organiser le travail prévu sur les ressources disponibles, et de maîtriser les conséquences des aléas internes sur des programmes de fabrication de plus en plus tendus. Un plan de fabrication doit, être flexible en fonction des données courantes de la production, des stocks et de la demande.

Dans la littérature, plusieurs modèles graphiques (les automates à états finis [6], les réseaux de Petri [7] ou encore les réseaux de files d'attente [8]) et analytiques (algèbre des dioïdes [9]) ont été proposés pour représenter les systèmes à évènements discrets y compris les systèmes HVLV qui mettent en jeu des phénomènes de synchronisation sans tenir compte des phénomènes de concurrence. Plusieurs chercheurs ont proposé ensuite des heuristiques afin de résoudre le problème de conflit dans les systèmes HVLV qui comportent des phénomènes de concurrence [10–12]. D'autres travaux de recherche se sont intéressés à la planification de maintenance dans ce type de systèmes [13, 14].

Les travaux présentés dans cette thèse portent sur la résolution du problème d'ordonnancement dans un système manufacturier de type HVLV. Une première étude approfondie des différentes caractéristiques de ce genre de systèmes a conduit à élaborer un modèle d'ordonnancement analytique en utilisant l'algèbre (max, +). Ce modèle permet de décrire le comportement discret des systèmes HVLV en tenant compte de la date de besoin.

Afin de pouvoir caractériser les modèles d'ordonnancement, rappelons que les systèmes HVLV peuvent être classés selon deux grandes catégories :

1. Les systèmes HVLV à décision libre qui sont des systèmes dans lesquels on suppose que les problèmes de concurrence entre les produits nécessitant un traitement sur une même machine ne sont pas présents. Autrement dit, l'ordre des opérations sur les machines est supposé connu à l'avance. Il s'agit de systèmes mettant en jeu des phénomènes de synchronisation et de retard que l'on retrouve abondamment dans les systèmes de transport (synchronisation de bus [15], les systèmes informatiques [16], etc).
2. Les systèmes HVLV à décision non libre qui présentent des problèmes de concurrence entre les produits nécessitant un traitement sur une même machine. Autrement dit, l'ordre des opérations sur les machines n'est pas connu à l'avance lorsqu'il y a concurrence entre deux produits différents. Il s'agit dans ce cas des systèmes intégrant des phénomènes d'ordonnancement assez complexes.

Afin de considérer les deux catégories de systèmes HVLV, nous avons adopté en premier temps un modèle (max, +) linéaire permettant de décrire le comportement discret des systèmes HVLV à décision libre et sans maintenance. Dans un second temps, nous avons amélioré le modèle (max, +) adopté afin d'étudier les problèmes de concurrence sur les machines dans les systèmes HVLV à décision non libre sans et avec maintenance. Le modèle ainsi proposé est un modèle non linéaire dans le sens de l'algèbre (max, +).

La majeure partie de la littérature dédiée aux problèmes d'ordonnancement se place dans le contexte où les ressources nécessaires à l'exécution des tâches sont toujours disponibles. Or, dans le cas industriel, cette hypothèse n'est pas toujours vérifiée. En effet, les différentes ressources humaines et/ou matérielles peuvent être indisponibles. Les dates et les durées des indisponibilités sont connues dans certains cas : congés personnel, opérations de maintenance sur les machines, etc. La maintenance préventive est une solution afin d'améliorer la fiabilité des machines et ainsi pallier au problème des pannes qui peuvent survenir sur les machines d'une façon soudaine. Dans ces conditions, nous avons tenu compte de l'état de disponibilité des machines dépendant de l'activité de maintenance préventive périodique dans les systèmes HVLV. Deux cas sont abordés dans les modèles développés : d'abord le cas où les activités de maintenance sont répétitives périodiques (les périodes entre les différentes opérations de maintenance sont égales) et ensuite le cas où les opérations de maintenance sont flexibles périodiques (les périodes entre les opérations de maintenance sont différentes).

Outre l'introduction générale ce mémoire est structuré en quatre chapitres et une conclusion et des perspectives :

- Le premier chapitre, propose un survol de la modélisation et du pilotage (ordonnancement) des flux de production dans les systèmes manufacturiers de type HVLV. Une caractérisation de ce type de systèmes est abordée. Un outil de modélisation algébrique est choisi (l'algèbre max-plus). Nous justifions alors notre choix et mentionnons ses avantages vis-a-vis d'autres modèles proposés dans la littérature.
- Dans le deuxième chapitre, nous étudions d'abord le comportement linéaire des systèmes HVLV à décision libre dans l'algèbre (max, +). Ensuite, un modèle d'ordonnancement (max, +) non-linéaire est proposé sans et avec maintenance. Plusieurs techniques de commande issues de l'automatique conventionnelle (commande en boucle ouverte, commande prédictive, commande par modèle interne, etc.) sont appliquées afin de piloter par poursuite une certaine trajectoire (date de besoin) dans le cas des systèmes HVLV libres. Dans le cas des systèmes non libres, le problème de pilotage est vu comme un problème d'ordonnancement. Dans ce cadre, un problème d'optimisation sous contraintes est présenté. Deux critères de performance seront optimisés : le makespan C_{max} pour un délai de production (production totale maximale) et la somme des avances R tout en satisfaisant la production en juste-à-temps (date de besoin), aspect peu étudié pour ce type de problème.
- Dans le troisième chapitre, le problème d'ordonnancement des systèmes HVLV est abordé sans et avec maintenance. D'abord, un état de l'art sur l'ordonnancement des systèmes HVLV est présenté. Ensuite, le problème d'ordonnancement des systèmes HVLV sans maintenance est résolu. D'une part, des contraintes sur les variables de décision sont élaborées afin de générer des ordonnancements faisables. D'autre part, des conditions sur les dates de début des tâches de maintenance sont établies dans l'algèbre (max, +) afin de respecter la périodicité entre les différentes activités

de maintenance. Dans ce contexte, l'activité de production est planifiée en utilisant l'activité de maintenance comme une contrainte d'indisponibilité des machines (ordonnancement conjoint).

- Le dernier chapitre est une application de l'approche de pilotage proposée. Afin de valider cette approche, elle est appliquée d'abord sur un système HVLV organisé en job shop carré de dimension 6×6 puis sur un système HVLV plus complexe de taille 10×10. Les deux exemples sont pris dans la littérature afin de comparer les résultats et de montrer les améliorations apportées.

Le manuscrit se termine par une conclusion générale dans laquelle nous proposons une synthèse des différents apports présentés dans cette thèse ainsi que des perspectives de nos travaux futurs.

Chapitre 1

Aperçu général sur les systèmes HVLV

1.1 Introduction

La complexité sans cesse croissante des systèmes de production a fait de la modélisation et du pilotage un des moyens les plus répandus et, dans certains cas, le seul dont on dispose pour l'étude de ces systèmes. Pour cette raison, toute technique visant à améliorer l'efficacité de cette approche est d'un très grand intérêt.

Les systèmes manufacturiers à faible cadence et à grande variété de produits appelés systèmes manufacturiers de type HVLV (*High-Variety, Low-Volume*) sont mal ou très peu étudiés dans la littérature d'un point de vue de modélisation et de pilotage. Notre objectif au cours de ce premier chapitre est de poser le contexte de notre recherche où les systèmes manufacturiers de type HVLV y sont présentés ainsi que leurs caractéristiques principales.

Nous commencerons ce chapitre par un survol de la littérature spécifique sur les SdPs (Systèmes de Production). Nous mettons d'abord l'accent sur les différentes parties d'un SdP, leurs objectifs ainsi que leurs caractéristiques. Ensuite, nous donnons une classification des SdPs pour pouvoir caractériser les systèmes HVLV et par la suite les représenter par un "bon" modèle décrivant au mieux leur comportement dynamique.

Après la présentation des différentes caractéristiques des systèmes HVLV, nous abordons le volet de l'évaluation et de l'analyse de leurs performances à l'aide de deux approches de modélisation, à savoir la méthode qualitative et la méthode quantitative. Une étude des propriétés de chaque technique de modélisation nous a permis de choisir l'outil le mieux adapté pour la modélisation des systèmes HVLV, à savoir l'algèbre (max, +). Nous introduisons par la suite le problème de pilotage (vu comme un problème d'ordonnancement). Nous finissons ce chapitre par la présentation des différentes raisons pour lesquelles nous avons choisi l'algèbre (max, +) comme outil de modélisation et de pilotage.

1.2 Les systèmes de production (SdPs)

Un système de production (SdP) est un espace constitué de produits (P), de moyens de production (M) et d'opérateurs (O) en interaction entre eux. L'établissement, le contrôle et l'exploitation des liens entre eux sont du ressort des outils de pilotage. Les événements associés aux SdPs sont situés dans l'espace (P, M, O, temps). La tendance actuelle s'oriente vers des systèmes produisant des petites et moyennes séries capables de s'adapter aux changements de production, afin de répondre aux exigences de diversité, de productivité et de qualité sollicitées par la concurrence actuelle du marché. Ces systèmes, appelés Systèmes Flexibles de Production Manufacturière (SFPM), y compris les systèmes HVLV (*High-Variety, Low-Volume*), se situent parmi les Systèmes Automatisés de Production (SAP). Les SFPM ont des installations manuelles à très faible automatisation permettant une production unitaire. Leur but est de fabriquer et de transformer une matière première en produit fini ou semi-fini. Un schéma simplifié des systèmes de production est représenté par la figure 1.1. Un SdP se décompose en deux parties complémentaires [17], la partie opérative et la partie pilotage.

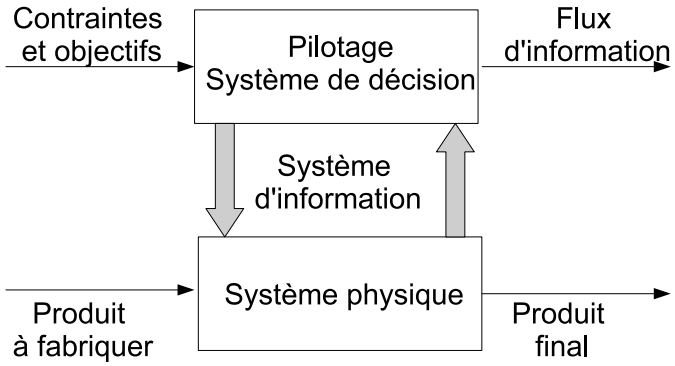

Figure 1.1 – Schéma d'un SdP [2]

1.2.1 Partie opérative

La partie opérative ou système de fabrication met en jeu de manière conventionnelle des entités appartenant à trois populations [18] : la population des moyens de production, la population des produits et la population des opérateurs de production.

- Les moyens de production regroupent les ressources physiques de l'atelier à savoir les ressources de traitement ou les machines et les ressources de transport qui assurent la manutention entre les ressources.
- Les produits qui sont caractérisés par leur gamme de fabrication ou la gamme opératoire. Cette dernière représente la suite ordonnée d'opérations à effectuer pour fabriquer ce produit. Chaque opération peut être exécutée par une ou plusieurs machines suivant un temps opératoire donné. Nous introduisons la notion de routage comme la suite ordonnée des moyens de production que visite le produit durant sa

fabrication.

- Les opérateurs de production représentent les ressources humaines.

1.2.2 Partie pilotage

La partie pilotage a pour rôle d'élaborer les décisions nécessaires à la partie opérative, de guider et piloter afin de satisfaire les objectifs de production. Les décisions portent sur les actions à mener par le procédé sur le produit. Ces actions sont élaborées à partir des informations (comptes-rendus) recueillies sur le procédé ou sur le produit. Dans [19], les auteurs identifient deux niveaux de décision :

- Niveau 1 : La gestion prévisionnelle, qui anticipe la programmation d'un ensemble de décisions afin de satisfaire les besoins des clients.
- Niveau 2 : Le pilotage, qui développe les décisions en temps réel.

1.3 Objectifs d'un système de production

Aujourd'hui, l'entreprise est confrontée à des changements importants et quotidiens mettant les systèmes de production face à des contraintes et des enjeux (voir figure 1.2) participent à une importante mutation de la fonction de production

Les systèmes industriels actuels doivent assurer une meilleure productivité avec un coût moins élevé. Les ingénieurs doivent donc concevoir des systèmes de plus en plus flexibles permettant un changement rapide de la production et des temps d'arrêt (entretien, maintenance) de moins en moins longs. Les SAP doivent donc répondre à des objectifs économiques, humains, techniques et de pilotage [20].

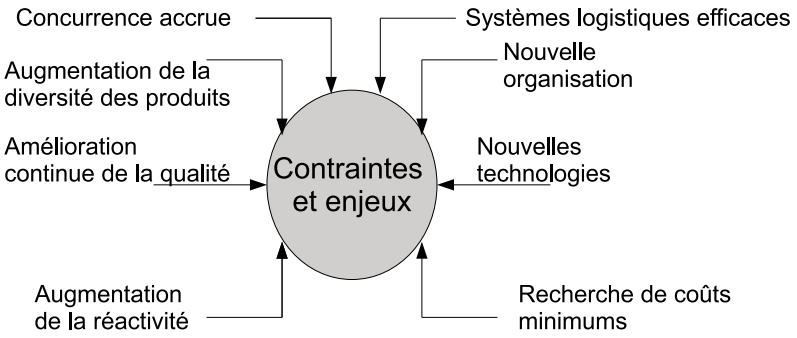

Figure 1.2 – Contraintes et enjeux [2]

1.4 Classification des Systèmes de production

Dans cette section nous allons classer les systèmes de productions en premier lieu selon le mode de pilotage. Ensuite, selon leur nature et le volume des flux physiques et enfin selon leur organisation logique (cheminement des flux).

1.4.1 Classification selon le mode de pilotage [1]

La première classification des systèmes de production est étroitement liée à la stratégie de pilotage utilisée. En effet, en se situant au niveau opérationnel et sur la base du mode de déclenchement de la production, cette classification sépare les systèmes fonctionnant à flux tirés de ceux fonctionnant à flux poussés [21] :

Les systèmes à flux tirés : La production est déclenchée par la consommation des produits finis. D'une manière générale, on peut distinguer deux types de fonctionnement. Le premier consiste à maintenir un stock minimum spécifié de produits finis. Dans ce cas, on parle d'une production sur stock (make to stock). Autrement dit, si un produit quitte le stock, un ordre de fabrication est lancé au système afin de pouvoir le reconstituer. Dans le deuxième type, la production est déclenchée par la réception d'une commande. En d'autres termes, ce type de fonctionnement vise à maintenir un stock de produits finis nul.

Les systèmes à flux poussés : Le déclenchement de la production est basé sur des planifications et des prévisions pour déterminer un programme de production. Dans ce type de gestion des flux, c'est la disponibilité du produit venant de l'amont qui déclenche l'étape suivante de fabrication. Cette méthode de production implique souvent le stockage des produits finis avant leur livraison.

Si d'un point de vue théorique cette classification reste pertinente, son déploiement dans la réalité industrielle est moins viable. En effet, dans la pratique, des systèmes de pilotage purement à flux tirés ou purement à flux poussés sont rares. Dans ce contexte, au niveau opérationnel, ces modes coexistent où un pilotage hybride à flux poussés et tirés est utilisé. Par exemple, dans un système de production, les produits simples sont fabriqués en flux poussés et les produits complexes sont assemblés en flux tirés.

1.4.2 Classification selon la nature et le volume des flux

Cette classification est basée sur la nature du système physique et la cadence des produits fabriqués par ce dernier. Dans ce contexte, on distingue principalement trois types de systèmes :

Systèmes à flux continu (SFC) : Les produits sont fabriqués en continu au cours d'un processus unique (ex : un verre, du ciment, ...). Les outils de production, fortement automatisés, sont implantés en ligne et permettent une circulation linéaire des matières premières. Les opérateurs sont peu nombreux et ils sont chargés de la surveillance et de l'entretien des équipements. La production s'effectue en continue (souvent 24h/24h et 7 jours/7). Dans ces systèmes, la matière circule en flux continu. C'est le cas des industries dites « process » où la matière est sous forme liquide ou gazeuse.

Systèmes à flux discret (SFD) : Dans ce genre de systèmes, les produits sont fabriqués en discontinu. Les quantités sont généralement restreintes et les produits sont variés.

Les outils de production sont moins spécialisés et ils sont regroupés par nature (ex : atelier découpe, atelier ponçage, etc). Les opérateurs sont généralement plus spécialisés. Dans ces systèmes, les produits sont distingués individuellement (production discrète). C'est le cas des industries manufacturières où trois classes peuvent être répertoriées :

- Les systèmes de production à grande cadence ou de masse : les produits sont lancés en grands volumes. Ces derniers passent par les mêmes ressources (machines, stocks. . .). Dans ce genre de systèmes, les biens sont produits en série. Il s'agit généralement de produits standardisés, fabriqués en grande quantité, pour réaliser des économies d'échelles (réduction des coûts de production du fait des quantités). La taille des séries est en fonction de la demande des clients et des coûts. La taille optimale sera celle qui permet de répondre à la demande des clients avec un coût minimum. Lorsque la série porte sur de grandes quantités (plus de 100 000 unités), on parle alors de production de masse (ex : la production de DVD vierges.).
- Les systèmes de production en moyenne série : contrairement à la classe précédente, il s'agit ici d'ateliers dans lesquels la diversité des produits ne permet pas une spécialisation des moyens de production. Les différents produits suivent leur propre chemin sur des ressources communes souvent regroupées par fonctionnalités équivalentes.
- Les systèmes de production unitaire appelés aussi les systèmes HVLV : Pour ce type de systèmes, la taille de produit ou la demande impose une production de très faible quantité avec une grande variété de produit. Dans ce cadre, les biens sont produits à l'unité. Il s'agit généralement de produits spécifiques répondant à des besoins particuliers qui font l'objet d'un cahier des charges détaillé. Il peut s'agir de produits très volumineux ou très coûteux, pour lesquels on parle de production unitaire par projet (ex : viaduc de Millau, usine clé en main, etc) ou de produits très personnalisés, pour lesquels on parle de production unitaire à la tâche (ex : conception d'un logiciel sur-mesure pour une entreprise donnée, conception d'un meuble sur-mesure, etc).

  Ce genre de systèmes est connu par des temps opératoires relativement longs ce qui prouve leur comportement discret et par des changements de série dûs à la variété de produits et par une production qui ne suit pas forcément le *takt time*. Ce sont des systèmes à évènements discrets (SEDs). C'est cette catégorie de systèmes qui nous intéresse dans notre étude.

Systèmes à flux hybride (SFH) [1] : Ces systèmes se situent à mi-chemin entre les systèmes à flux continu (SFC) et les systèmes à flux discret (SFD). Deux configurations peuvent être distinguées :

- Les deux types de systèmes (continu et discret) sont couplés : la production est continue tout en ayant un conditionnement discret des produits.
- Les deux aspects continu et discret cohabitent : dans le même système de production, les traitements sont continus mais effectués par lots.

Dans certains cas, les deux configurations précédentes peuvent être associées. Pour une étude plus détaillée sur les systèmes hybrides (discontinus), le lecteur peut se référer par exemple à la thèse de Andreu [22].

1.4.3 Classification selon la nature de la demande

Cette classification consiste à distinguer les systèmes où la production est déclenchée par les commandes des clients, de ceux dont la production s'effectue sur une anticipation de ces demandes [23]. Dans ce dernier cas, les produits sont stockés en vue de satisfaire une commande ultérieure. Généralement, on distingue :

- Les systèmes basés sur une production à la commande : ils concernent principalement les entreprises proposant une grande variété de produits, dont la demande est très aléatoire, ou celles qui ne définissent leurs produits qu'à partir de demandes typiques.
- Les systèmes basés sur une production sur stock : ce type de production n'est possible que pour des entreprises dont la gamme de produits proposés est peu évolutive, et surtout dont la demande sur ces produits est suffisamment prévisible. De plus, pour être réellement intéressant, un tel type de production doit se justifier par des pics de demande ne nécessitant pas à eux seuls, une augmentation de la capacité de production. Par exemple, la fabrication d'automobiles ne peut s'effectuer sur demande, mais doit devancer celle-ci (prévisions).

1.4.4 Classification selon l'organisation logique des moyens

D'une manière générale, selon le routage de produits à travers les machines et l'ordre des opérations, on distingue :

Les organisations à flots (Flow-Shop) : On peut représenter deux type d'ateliers flow-shop :

- Flow-shop classique : on le rencontre dans les ateliers de production de masse avec peu de variété de produits. Un tel atelier est aussi appelé « atelier à cheminement unique » où toutes les gammes sont identiques. Ainsi, l'ordre de passage des opérations sur les machines est le même pour tous les jobs (voir figure 1.3).

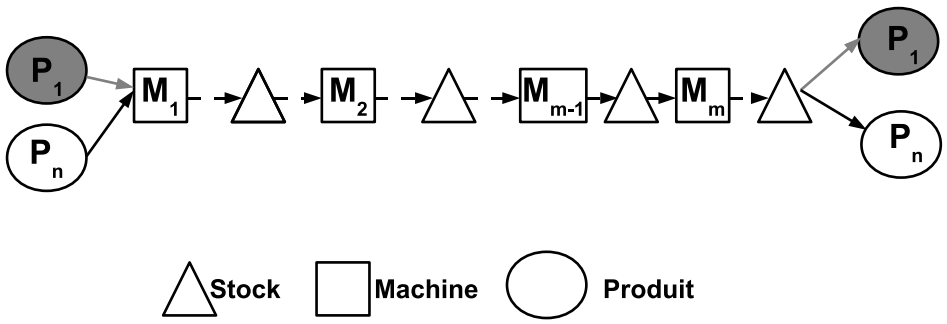

Figure 1.3 – Flow-Shop

- Flow-shop généralisé : présent dans les ateliers de production de faible cadence et à grande variété de produits. Un tel atelier est aussi appelé « atelier à cheminement multiple » où toutes les gammes sont différentes. Selon leur type, les produits ne passent pas forcément par toutes les machines (voir figure 1.4). Les flow-shop généralisés sont des systèmes HVLV.

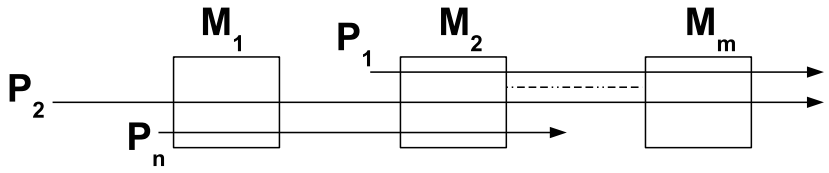

Figure 1.4 – Flow-Shop généralisé

Les organisations à tâches (Job-Shop) : Dans cette classe d'ateliers, chaque tâche possède son propre routage sur les machines (voir figure 1.5). Ce type d'atelier est dédié à une production en grande variété de produits, plus particulièrement aux systèmes HVLV [24].

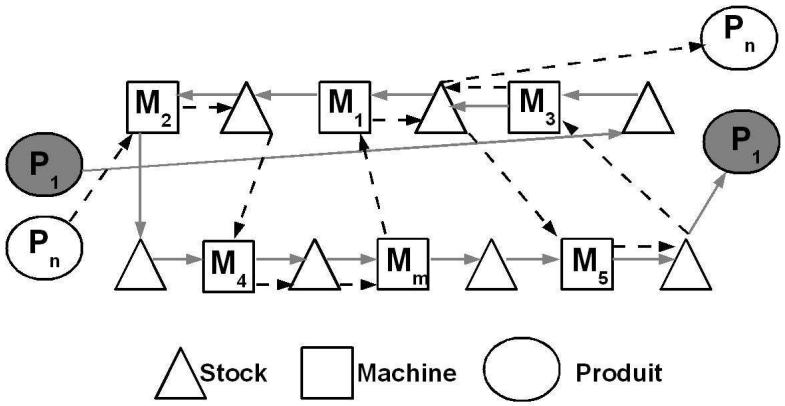

Figure 1.5 – Job-Shop

Les organisations à cheminement multiple (Open-Shop) : dans ce cas, le routage des jobs (produits) est multiple, mais à la différence du Job-Shop, les jobs ne possèdent pas de gammes. L'ordre de passage des opérations est ainsi quelconque (atelier à cheminement libre). Ce paramètre est déterminé lors de l'ordonnancement. Le routage est vu comme la suite ordonnée des moyens de production que visite le produit durant sa fabrication.

1.4.5 Discussion

La classification des systèmes abordée dans cette partie est résumée sur la Figure 1.6. Selon la première classification (selon le mode de pilotage), nous nous intéressons à la

production à flux discret et plus spécifiquement à la production en faible série (faible volume). En effet, c'est dans cette catégorie de production que les problèmes de pilotage des flux et, plus particulièrement, les problèmes d'ordonnancement sont les plus complexes. De plus, la densité des flux des produits circulant dans ces systèmes nous motive à étudier le problème de traçabilité des différents types de produits sur les machines via certaines de leurs attributs, à savoir leurs **dates de besoin**. Par conséquent, une approximation de flux de produits par un modèle « fluide continu » [1] semble limitée dans le cas des systèmes à une faible cadence de production. Une modélisation continue par des équations différentielles [1] n'est pas adéquate pour représenter les systèmes HVLV. En effet, la modélisation continue ne tient pas compte de l'aspect discret des flux de production et n'est pas capable de traiter des problèmes de concurrence et de synchronisation. D'où la nécessité d'une modélisation discrète qui tient compte des attributs du système manufacturier.

En fonction de la deuxième classification (selon la nature et le volume des flux physiques), nous nous situons dans le cadre d'un pilotage où les deux modes de gestion des flux poussés et tirés sont présents. La problématique de pilotage liée à cette classification fera l'objet des chapitres suivants. Dans la suite, nous nous focalisons sur les systèmes HVLV vus comme des systèmes de production à flux discret en détaillant les différentes techniques de modélisation de ces systèmes.

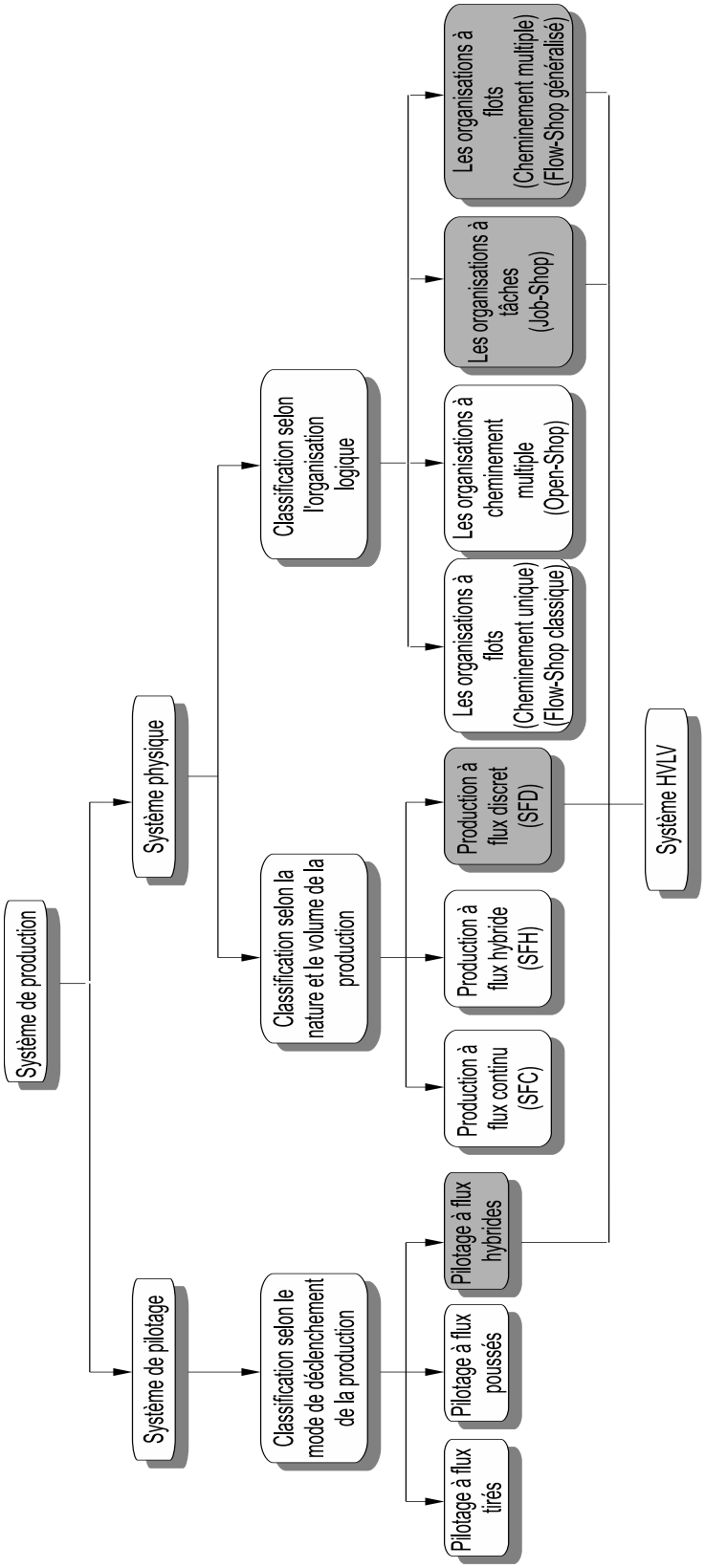

Figure 1.6 – Classification des systèmes de production

1.5 Modélisation des systèmes HVLV

Cette section présente les concepts utilisés dans cette thèse pour présenter la méthodologie de modélisation des flux dans un système HVLV. Nous mettons en évidence l'apport d'une modélisation discrète dans ces systèmes. Nous abordons le volet de l'évaluation et de l'analyse des performances à l'aide de deux méthodes : méthode qualitative et méthode quantitative.

D'une manière générale, la modélisation d'un système de production est un problème complexe. Disposer d'un « bon modèle » pour un système donné apporte une aide indispensable aux concepteurs ainsi qu'aux utilisateurs. Ce modèle va permettre au système, selon l'objectif visé, de faire de la simulation, de l'évaluation de performances, d'optimiser le fonctionnement et de tester différentes politiques de pilotage.

Comme nous avons indiqué dans la section 1.4.2, les systèmes HVLV sont des Systèmes à Évènements discrets (SEDs). Dans la littérature, nous trouvons deux méthodes de modélisation des SEDs : une méthode qualitative et une méthode quantitative. Dans ce contexte nous commençons d'abord par une définition générale des termes « système » et « modèle ». Ensuite nous définissons les SEDs. Enfin nous exposons les deux méthodes de modélisation afin de choisir la méthode la plus appropriée pour représenter le comportement complet de notre système.

1.5.1 Système et modèle

Définition d'un système

Un système est un « ensemble d'éléments en interaction mutuelle et en interaction avec l'environnement, organisés en fonction d'un même but pour parvenir à une même fin » [25]. Il convient d'ajouter qu'un système possède une structure dynamique : son bon fonctionnement dépend de la bonne synchronisation entre ses processus.

Définition d'un modèle

Un modèle est une représentation simplifiée capable de reproduire de façon pertinente le comportement partiel ou total d'un système. Il existe trois types de modèles (voir figure 1.7) :

- Les modèles mathématiques tels que les équations aux dérivées partielles et l'algèbre des dioïdes (mécanique des fluides, géologie, météorologie, systèmes de transport, systèmes manufacturiers, océanographie,etc).
- Les modèles fondés sur des méthodes d'apprentissage tels que les réseaux de neurones artificiels (traitement de signal, contrôle de processus, classification de données, etc).
- Les modèles graphiques tels que les Réseaux de Petri (RdPs) (systèmes de transport, systèmes manufacturiers, etc).

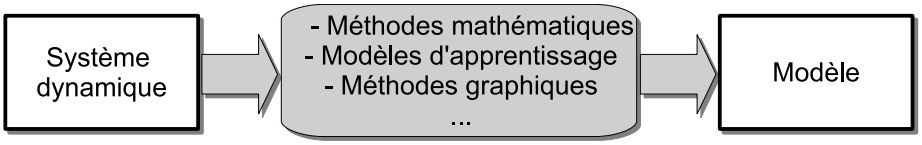

Figure 1.7 – Modélisation d'un système dynamique

1.5.2 Les systèmes à évènements discrets (SEDs)

La théorie des systèmes continus [26, 27] et de l'automatique s'intéresse à des systèmes « naturels » obéissant essentiellement aux lois de la physique et descriptibles par des équations différentielles ou par des dérivées partielles (ou leur discrétisation approchée en temps). Cependant, la théorie des SEDs (voir figure 1.8) recouvrent des systèmes également dynamiques, mais dont la dynamique échappe totalement à ce genre de description. En réalité, c'est plutôt le niveau descriptif auquel on se place qui est la source de cette impossibilité. En effet, au lieu de s'intéresser au déroulement continu des phénomènes, on ne se soucie que des « débuts » et des « fins » de ces phénomènes (les événements discrets) et de leur enchaînement dynamique, logique ou temporel. Il s'agit de systèmes, dont le comportement peut être représenté sous forme d'occurrences asynchrones d'événements discrets. Ces occurrences peuvent être contrôlées ou non-contrôlées. Un événement discret peut être défini comme étant un changement qualitatif d'une situation. A titre d'exemple on peut citer la fin d'une opération dans un atelier de production, l'apparition d'un nouvel objet dans une zone donnée ou le déclenchement d'une alarme.

Formellement, un SED est un système, ou plutôt une vision d'un système, pour laquelle le temps et les composantes du vecteur d'état sont des variables discrètes.

Les SEDs peuvent être rencontrés dans plusieurs domaines : la production industrielle, le transport, le domaine de l'informatique, la communication,etc. Classiquement, lorsque l'on cherche à étudier un système manufacturier, on le considère comme un SED ; c'est pourquoi nous allons nous s'attarder à sa description.

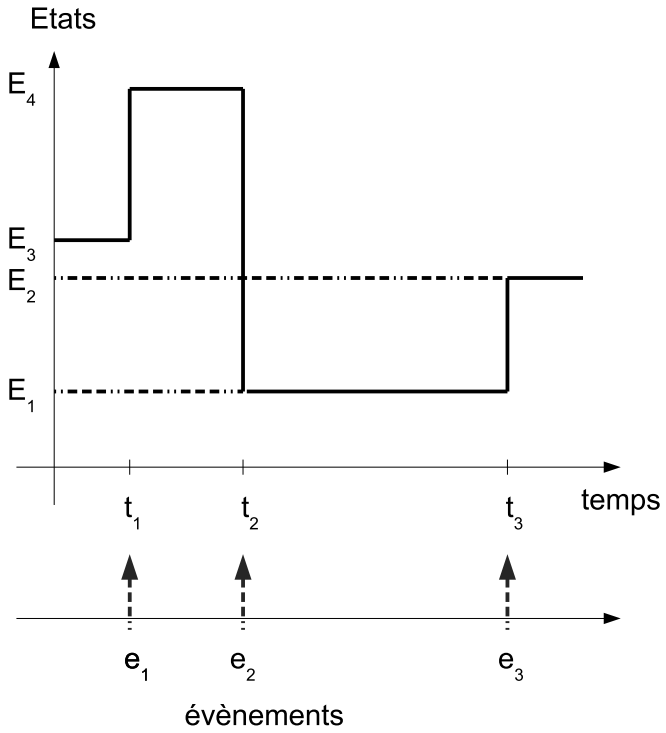

Figure 1.8 – Évolution des états dans un SED

Les SEDs recouvrent plusieurs domaines d'application dont les systèmes manufacturiers de type HVLV. Les SEDs sont des systèmes dynamiques qui évoluent en fonction de l'occurrence brutale d'un évènement physique e_i dans des intervalles de temps irréguliers $[t_{i-1}, t_i]$. Pour chaque évènement qui change, lui correspond un nouveau état E_i du système. La durée de temps pendant laquelle le système persiste dans un tel état jusqu'à l'apparition d'un nouvel évènement est appelé « temps de maintien dans un état ». La trajectoire (évolution du système) est caractérisée par la séquence des états et des « temps de maintien » dans ces états [28].

Bien qu'il y ait eu plusieurs tentatives pour établir un modèle général pour les SEDs, il n'y a pas encore un cadre "parfait" de modélisation comme dans le cas des systèmes continus [26, 27].

Lorsque l'entrée du SED est spécifiée de manière déterministe comme une séquence d'événements , sans information sur le temps auquel les occurrences de ces événements se produisent, partant d'un état initial, on décrit la trajectoire en termes de séquences d'états résultants. On parle dans ce cas d'un **modèle non temporisé ou d'un modèle qualitatif** [29]. Ce type de modèle permet de répondre en particulier à la question : « Un état particulier peut-il être atteint ? »

Dans ce cas, le système évolue autour d'un nombre limité d'états et ce modèle n'est pas convenable pour des systèmes dont le nombre des états physiques discrets "explose" d'une façon combinatoire [28].

Lorsque l'entrée du SED est spécifiée de manière déterministe comme une séquence d'événements auxquels sont associées les dates d'occurrence t_i, on peut construire la tra-

jectoire complète du **modèle temporisé ou d'un modèle quantitatif**. Les modèles temporisés tels que les chaînes de Markov, les réseaux de files d'attentes et l'algèbre des dioïdes sont utilisés pour répondre à des questions quantitatives et de performances (débit du système, *Work-In-Process : WIP*, date de besoin, temps de séjour moyen,...). Ils répondent notamment aux questions :

- Quand un état particulier sera-t-il atteint ?
- Combien de temps le système passe-t-il dans un état particulier ?
- Combien de fois un état particulier peut-il être atteint sur un intervalle de temps donné ?

1.5.3 Modélisation qualitative et quantitative des SEDs

Rappelons qu'un modèle est une approximation, une vue partielle plus ou moins abstraite de la réalité afin de l'appréhender plus simplement, selon un point de vue et qu'il est établi pour un objectif donné. Un modèle est donc subjectif puisqu'il est établi en fonction des objectifs, du jugement, de la nature et de la qualité des informations dont dispose le concepteur. Il peut être exprimé par des mathématiques, des symboles, des mots,... [30]. Dans la suite seront présentés succinctement les réseaux de Petri, les langages et automates, l'algèbre (max,+) trois outils de modélisation et d'analyse des systèmes qui trouvent également une application en commande.

Modélisation qualitative

Réseaux de Petri (RdPs) non temporisés : Les réseaux de Petri (RdP), qu'ils soient ordinaires, généralisés, colorés, temporels et stochastiques sont un outil graphique à support mathématique permettant de modéliser, visualiser et analyser des évolutions comportant du parallélisme, de la synchronisation et du partage de ressources. Un réseau de Petri (RdP) est un moyen de modélisation du comportement des systèmes dynamiques à événements discrets et de description des relations existantes entre des conditions et des évènements [7, 31–33]. Les RdPs ordinaires non marqués, les RdPs généralisés non marqués, les RdPs marqués et les RdPs synchronisés appartiennent tous à la classe des RdPs non temporisés.

Langages et Automates : Les langages et les automates permettent de traiter mathématiquement les problèmes relatifs aux SED, essentiellement d'un point de vue logique (analyse qualitative). La théorie des automates à états finis (à nombre d'états fini) a été principalement développée avec la théorie des langages. Ces modèles reviennent à spécifier des ensembles d'états et des transitions entre ces états [34]. Chaque SED a un ensemble d'événements qui lui est associé, ces événements font évoluer le SED. Cet ensemble peut être vu comme un alphabet d'un langage et les séquences d'événements sont des mots (aussi appelés chaînes) de ce langage. Un automate est alors un dispositif qui engendre un langage en manipulant l'alphabet (les événements).

Modélisation quantitative

Réseaux de Petri (RdPs) temporisés : La théorie originale des RdPs traite principalement la chronologie et le séquencement des événements. Toutefois, pour les questions liées à l'évaluation des performances (à quelle vitesse peut produire un réseau ?), il est nécessaire d'introduire le temps. Cela peut être fait de deux manières fondamentales en associant des durées des tirs des transitions ou avec le séjour de jetons dans les différentes places du réseau.

Les durées associées aux tirs des transitions peuvent être utiles pour représenter les temps de production dans un SdP, où les transitions représentent les machines, la longueur d'un code (programme) dans le cadre informatique,...

Un RdP P-temporisé (respectivement T-temporisé) est un 2-uplet $\langle R, Tempo \rangle$ tel que :

- R est un RdP marqué ;
- Tempo est une application de l'ensemble P des places (respectivement de l'ensemble T des transitions) dans l'ensemble des nombres rationnels positifs ou nuls. $Tempo(P_i)$ est la valeur de la temporisation associée à la place P_i (respectivement $Tempo(T_j)$ est la valeur de la temporisation associée à la transition T_j) [35, 36].

Il existe des RdP P-temporels (respectivement T-temporels), pour lesquels on associe aux places (respectivement aux transitions) une temporisation dont la valeur se situe à l'intérieur d'un intervalle. Cette situation se rencontre typiquement pour les bains de trempage ou les fours pour lesquels une pièce doit séjourner entre une durée minimale et une durée maximale.

Algèbre des dioïdes : Le mot dioïde désigne une structure algébrique analogue à un anneau. Dans un dioïde, l'opération max représente la contrainte de synchronisation et l'addition usuelle sert à représenter les décalages temporels.

Cette approche proposée par Cohen et *al.* en 1983 [37, 38] a été confrontée aux outils de modélisation graphiques dont l'usage commençait à se généraliser, à savoir les réseaux de Petri et les Graphes à Évènements Temporisé (GET). Cette interface avec les outils graphiques a été complétée par une études des propriétés algébriques des dioïdes dont la première loi (addition) est issue d'une relation d'ordre. La structure algébrique de dioïde permet de modéliser et d'évaluer les performances de certains systèmes à événements discret [39,40]. Une représentation d'état linéaire de cette classe de systèmes peut être établie. Il est possible par la suite de déterminer efficacement leur comportement périodique [9].

Deux types de représentation d'état linéaires dans l'algèbre de dioïde peuvent être associés aux graphes d'évènements temporisé. La première formulation calcule le temps physique (dates des évènements) comme fonction du temps logique (numérotation des évènements) et utilise le dioïde (max,+). La seconde formulation calcule le temps logique comme fonction du temps physique, et utilise le dioïde (min,+). Ces deux formulations sont duales et strictement équivalentes : aucune n'apporte plus d'informations que l'autre. Les matrices et coefficients sont directement issus de la topologie et de la temporisation du réseau de Petri ainsi que de son marquage initial.

Plusieurs travaux se sont développés par la suite pour modéliser les systèmes qui peuvent être représentés par des GET afin d'obtenir un modèle linéaire invariant [36, 41–43]. Ils ont présenté un modèle pour l'analyse et l'évaluation des performances des SEDs et ils ont fourni une excellente revue de l'algèbre des dioïdes. Cohen et *al.* [44] ont proposé une approche pour l'analyse spectrale des GET en utilisant l'algèbre des dioïdes ainsi que le traçage du diagramme de Bode d'une manière analogue à celle des systèmes linéaires dans l'automatique continue. Une représentation formelle (fonction de transfert) a été établie par la suite pour les GET à partir de la forme linéaire. Cela a nécessité la résolution de quelques difficultés « techniques » liées à l'algèbre des dioïdes.

L'utilisation de l'algèbre des dioïdes pour la modélisation des systèmes manufacturiers a été initiée par Cohen et *al* [45]. Dans ce cadre, un modèle d'état qui consiste en un ensemble d'équations récurrentes était obtenu. Cohen et *al.* ont montré que les SEDs, (les systèmes manufacturiers), sont linéaires dans le sens de l'algèbre des dioïdes. Il est important de noter que cette algèbre était initialement utilisée pour modéliser les systèmes à décision libre où l'ordre des opérations des différents produits traitées sur la même machine est supposé connu d'avance. Les équations linéaires dans les dioïdes modélisent uniquement les problèmes de synchronisation. Dans ce contexte, Cohen et *al* [45] ont défini des concepts de stabilité, de périodicité et de valeurs propres comme dans le cas des systèmes continus. L'algèbre des dioïdes a été appliquée pour la résolution des problèmes de graphes et réseaux [40]. Dans le cas des systèmes qui présentent des phénomènes de conflit d'allocation des ressources, les modèles linéaires dans les dioïdes ne sont plus appropriés. Pour pouvoir appliquer le calcul linéaire dans les dioïdes, il faut réduire la complexité combinatoire du problème, en effectuant des choix d'allocation de ressources dans le système.

Dans [44,46,47], une symétrisation de l'algèbre (max, +) et une nouvelle notion appelée « *balance* » a été introduite. Cette technique permet la généralisation des équations classiques. Les auteurs ont montré que chaque système non dégénérant des balances linéaires a une solution unique dans l'algèbre (max, +) en utilisant la règle de Cramer.

Une autre contribution très importante a été proposée dans l'algèbre des dioïdes par Resing et *al.* [48] et Olsder et *al.* [49]. Ces derniers ont prouvé que l'algèbre (max, +) peut modéliser les SEDs stochastiques avec des temps de production et de transport subissant des fluctuations stochastiques. Resing et *al.* [48] ont étudié des SEDs stochastiques, i.e, les éléments de la matrice qui décrit la dynamique du système et le vecteur d'état sont stochastiques.

1.6 Pilotage des systèmes HVLV

1.6.1 Contexte de notre travail

Le pilotage de toutes les grandes manufactures modernes, y compris les systèmes HVLV, implique un système de surveillance étendu caractérisé par un grand nombre de variables opérant. Un tel système exige le développement d'une multitude de relations assez complexes, souvent non linéaires afin de déterminer les actions de contrôle nécessaires

à partir de l'ensemble des variables courantes de processus. En plus, ces systèmes sont toujours sujets à des problèmes continus de réglage des calendriers de leur production afin de satisfaire les besoins de leurs clients tout en essayant d'ajuster leur productivité avec les moindres coûts.

Le contrôle des opérations dans un SdP est normalement effectué par un système informatique intégré. Dans le cas industriel, le problème de pilotage d'un système HVLV est beaucoup plus complexe dû aux variabilités qui peuvent exister dans ce genre de systèmes (pannes des machines, variabilité de la demande client,...). En plus, l'aspect discret des opérations de production dans les HVLV ainsi que la variété de produits rend plus difficile la détermination d'actions de pilotage optimal [50].

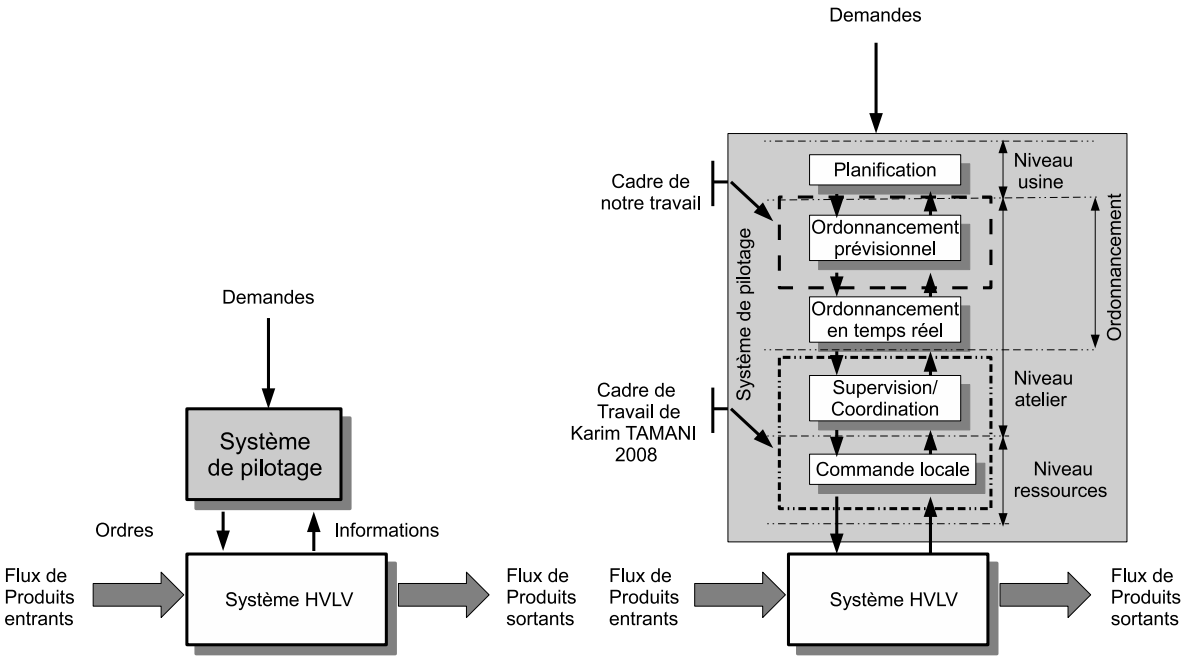

Figure 1.9 – La conduite hiérarchisée des systèmes HVLV

Dans le cadre du pilotage des systèmes de production, une vision globale à cinq couches couramment utilisée dans la littérature est illustrée sur la figure 1.9 Le rôle de chaque niveau est défini de la façon suivante [1, 51] :

- La planification : elle consiste à définir un plan de fabrication sur un horizon de temps en fonction du carnet de commande fourni par le client et des moyens de production. Ce plan fait apparaître des valeurs comme la quantité, la qualité, les délais de production et également une pré-allocation des ressources.
- L'ordonnancement prévisionnel : il permet de déterminer un ensemble d'ordres partiels de passage des produits à transformer sur les diverses ressources du système. Cette allocation tient compte de la pré-allocation des ressources, des délais, des contraintes de capacité et de disponibilité des ressources, des contraintes de séquencement d'opérations et enfin des critères à optimiser tels que le temps de production, les coûts,...

- L'ordonnancement temps réel : il assure la cohérence entre les décisions prévisionnelles et les contraintes temps réel issues du comportement réel du système. Ce niveau est souvent qualifié de « charnière » puisqu'il permet de gérer, en fonction de l'état réel des ressources, au mieux les degrés de liberté non encore explicités par les niveaux supérieurs.
- La supervision/coordination : elle permet de gérer de manière cohérente les interactions entre les différentes ressources du système en fonction des contraintes telles que les ressources partagées, séquencements obligatoires, synchronisations diverses ou parallélisme [52], etc. De par sa vision globale du système, le superviseur permet d'introduire des informations globales au niveau de la commande locale afin d'optimiser les différentes opérations du système physique.
- La commande locale : elle correspond au niveau le plus bas de la structure décisionnelle. C'est la jonction entre le procédé et le système de commande. La commande locale consiste en un enchaînement déterministe d'opérations à effectuer sur les ressources du système physique. Ainsi, la commande locale intègre des fonctions de poursuite et de régulation.

Le schéma de la figure 1.9 peut être scindé en deux classes. La première classe regroupant les deux couches supérieures (la planification et l'ordonnancement prévisionnel) concerne essentiellement des problèmes de pilotage en temps non critique. Dans ce cas, le problème de pilotage est vu comme un processus décisionnel à un niveau stratégique [52]. La deuxième classe, englobant les trois couches inférieures (ordonnancement en temps réel, supervision et commande locale) concerne un pilotage en temps réel où la proximité avec le système physique impose la prise en compte des contraintes réelles.

Nous allons nous intéresser dans cette thèse à la première couche supérieure, à savoir l'ordonnancement prévisionnel. Donc pour nous, la structure et le séquencement des tâches dans le système de production sont inconnus et à déterminer.

Dans ce cadre, notre approche se différencie des travaux proposés dans [1] où les auteurs supposent que le séquencement des tâches est connu et fixé a priori.

Dans ce cas, leur problème de pilotage en temps réel se limite à l'étude et à la conception des deux dernières couches, à savoir la commande locale et la supervision.

1.6.2 Les méthodes d'ordonnancement

La plupart des problèmes d'ordonnancement sont NP-difficiles. Pour les résoudre, il existe principalement deux méthodes : les méthodes exactes (si la nature et la taille du problème le permettent) et les méthodes approchées (si le temps de calcul pour trouver une solution exacte n'est pas admissible).

Méthodes exactes : On peut définir une méthode exacte comme une méthode qui garantit l'obtention de la solution optimale pour un problème d'optimisation. L'utilisation de ces méthodes s'avère particulièrement intéressante. Parmi ces méthodes, on peut citer

les méthodes de séparation et d'évaluation, de programmation dynamique et de programmation linéaire et non-linéaire.

La méthode par séparation et évaluation (*Branch and Bound*) : C'est une méthode basée sur une technique implicite énumérative. En effet, elle permet de trouver une solution optimale, en construisant un arbre de recherche, en examinant systématiquement les sous-chemins qui sont susceptibles d'aboutir à une solution réalisable et en excluant les autres sous-arbres de la recherche [53].

La programmation dynamique : C'est une méthode d'optimisation qui opère par phases. Son efficacité repose sur le principe d'optimalité de Bellman, à savoir « toute politique optimale est composée de sous-politiques optimales » [54]. Cette méthode permet une résolution ascendante. La solution optimale d'un problème est obtenue à partir des solutions de tous les sous problèmes.

La programmation linéaire (PL) : C'est une des techniques classiques de la recherche opérationnelle. Cette méthode repose sur la méthode du simplexe. Elle consiste à minimiser une fonction coût en respectant des contraintes. Les critères et les contraintes sont des fonctions linéaires des variables du problème.

La programmation non-linéaire (PNL) : Si moins une contrainte ou la fonction objective n'est pas une combinaison linéaire de variables d'optimisation, on parle alors de programmation non linéaire (PNL).

Méthodes approchées : Les problèmes d'ordonnancement étant en général NP-difficiles, on cherche à les simplifier pour réduire le volume des calculs. Bien entendu, cela se fait au prix d'une dégradation de la qualité de la solution. Il existe plusieurs familles de méthodes qui permettent d'approcher efficacement la solution optimale. Parmi ces méthodes on trouve les méthodes heuristiques et les métaheuristiques.

Les heuristiques sont des méthodes empiriques, qui donnent généralement de bons résultats, sans être démontrables mathématiquement. Elles se basent sur des règles simplifiées pour optimiser un ou plusieurs critères. Le principe général de cette catégorie de méthodes est d'intégrer des stratégies de décision (FIFO, EDD : *Earliest Due Date*, SPT : *Shortest Processing Time*,...) pour construire une solution proche de l'ordonnancement optimal, tout en essayant d'avoir un temps de calcul raisonnable [55].

Les métaheuristiques représentent des concepts généraux de résolution. Il faut formuler le problème abordé de telle manière qu'il soit adapté à l'application de ces concepts. Parmi les méthodes métaheuristiques on a :

- les méthodes par construction,
- les méthodes par décomposition,
- les méthodes par voisinage (recherche tabou, algorithmes génétiques,...).

1.6.3 Choix de la problématique et orientation de notre travail de recherche

Comme indiqué dans la section 1.5, les modèles pour les SEDs peuvent être classés en deux catégories majeures [56] : modèles temporisés et modèles non temporisés.

Les modèles non temporisés tels que les machines à états finis et les RdPs non temporisés mettent l'accent sur les séquences d'états et d'évènements du SED et ils ignorent complètement la durée de séjour du système dans chaque état ou évènement. Ces modèles représentent les SEDs d'une manière logique ou qualitative (accessibilité et bornitude [56]). Ces modèles deviennent plus difficiles à étudier et à analyser dans le cas de représentation des systèmes complexes tels que les systèmes HVLV (le nombre d'états physiques discrets explose d'une façon combinatoire) [28].

Le choix d'un type de modèle est fonction des objectifs recherchés et de la nature du système considéré. L'objectif de cette thèse est de développer une méthodologie de pilotage des systèmes HVLV qui sont des SEDs (nature du système) au sens du suivi de trajectoire « **date de besoin** » (objectif à atteindre).

Le comportement discret des systèmes HVLV peut être complètement caractérisé par la connaissance des dates de début et de fin de ces activités. La relation (relation d'ordre) entre ces différentes activités nécessite des opérations algébriques telles que le *maximum* et l'*addition* qui sont deux opérations de base de l'algèbre (max, +). Afin de satisfaire la demande du client dans les systèmes HVLV (date de besoin), des problèmes de concurrence et de conflit entre les opérations sur les mêmes machines doivent être abordés et résolus selon les priorités de chaque types de produits (date de besoin la plus urgente). A ce niveau, il s'agit de résoudre des problèmes complexes d'ordonnancement. Par conséquent, les caractéristiques de l'algèbre (max, +) lui permettent d'être un bon outil pour représenter le problème d'ordonnancement des systèmes HVLV. En effet, l'algèbre (max, +) est capable de calculer la commande en Juste-à-Temps (JAT) afin de satisfaire les dates de besoin.

Nous avons vu dans la section 1.5.3, que les modèles (max, +) linéaires sont seulement dédiés à traiter des problèmes de synchronisation dans les systèmes de production, y compris les systèmes HVLV [57]. Cette représentation linéaire dans l'algèbre des dioïdes permet d'appliquer des techniques de commande issues de l'automatique classique pour les systèmes continus [9, 45, 58–62].

Pour pouvoir utiliser l'algèbre (max, +) pour l'ordonnancement des systèmes HVLV, il est nécessaire d'introduire une stratégie de prise de décision dans le modèle afin de résoudre le conflit entre les opérations sur les machines. Nous allons montrer dans le troisième chapitre que ceci entraîne une non-linéarité dans le modèle (max, +) proposé.

Le problème d'ordonnancement traité dans ce travail consiste à trouver un ordre de passage des différentes opérations de produits sur l'ensemble des machines, en respectant toutes les contraintes telles que les interventions pour effectuer des tâches de maintenance (objectif du quatrième chapitre) tout en minimisant un certain critère de performance (makespan et avance totale en juste à temps).

Les industries de type HVLV doivent assurer la qualité d'une production à un coût minimum et dans un délai raisonnable, alors que la demande est de plus en plus variée et exigeante. Ceci impose aux gestionnaires d'atelier de réaliser un ordonnancement prévisionnel et d'en déduire des propositions des meilleures dates de livraison pour les commandes planifiées.

L'utilisation d'une méthode exacte pour résoudre le problème d'ordonnancement dans les systèmes HVLV semble intéressante. En effet, les méthodes exactes permettent d'avoir la valeur optimale du critère à optimiser, tandis que les méthodes approchées fournissent des valeurs qui sont au voisinage de l'optimal. En plus, l'utilisation des méthodes approchées nécessite souvent une expérimentation importante dans le sens de réglage des paramètres de convergence (les algorithmes génétiques) ou dans le sens du choix d'un algorithme le plus adéquat afin de converger vers la valeur optimale du critère.

Nous allons démontrer dans le chapitre 2 que les systèmes HVLV à décision non libre peuvent être représentés par un modèle (max, +) non-linéaire. D'après nos connaissances, il n'existe pas encore des techniques pour la résolution des modèles non-linéaire dans l'algèbre des dioïdes. Par conséquent, nous allons adopter, dans cette thèse, une méthode d'optimisation discrète non-linéaire sous contraintes (programmation non linéaire PNL) afin de résoudre le problème d'ordonnancement des systèmes HVLV tout en optimisant certains critère de performance.

1.7 Conclusion

Dans ce chapitre, nous avons introduit d'une manière générale les SdPs. Nous avons également présenté leurs structures, leurs objectifs et leurs caractéristiques.

Dans une deuxième partie, la classification des SdPs selon leur mode de pilotage, leur nature et le volume physique, la nature de la demande ainsi que leur organisation logique nous a permis de caractériser le système étudié dans cette thèse, à savoir les systèmes HVLV.

Dans une troisième partie, nous avons abordé le volet de la modélisation des SEDs (quantitative et qualitative). L'aspect discontinu des flux de production dans les systèmes HVLV, nous a motivé de choisir une représentation quantitative de ce type de systèmes. En effet, une modélisation quantitative en utilisant l'algèbre (max, +) tient compte des attributs du système (date de besoin, nombre de pièces,...).

La dernière partie de ce chapitre était consacrée à la présentation du contexte général du pilotage des systèmes manufacturiers. Nous avons défini les différentes couches de pilotage (la planification, l'ordonnancement, la supervision/coordination et la commande locale). Nous avons pu, par la suite, localiser le contexte de notre travail dans le cadre de l'ordonnancement des systèmes HVLV.

Dans le chapitre qui suit, nous allons proposer une modélisation par représentation d'état linéaire dans l'algèbre (max, +) d'une classe des systèmes HVLV, à savoir les systèmes HVLV à décision libre. Afin d'illustrer le fonctionnement et la technicité de l'ap-

proche (max,+), nous allons introduire et appliquer deux types de commande (max, +) (commande en boucle ouverte et commande en boucle fermée) afin de résoudre un problème de poursuite de trajectoire (date de besoin de produits) en juste-à-temps.

Chapitre 2

Modélisation et pilotage à base de (max,+) des systèmes HVLV

2.1 Introduction

L'objectif de ce chapitre est de proposer un survol sur la modélisation et la commande des systèmes HVLV. D'une manière générale, si les systèmes HVLV à décision libre peuvent être appréhendés comme des systèmes linéaires sur la structure algébrique (max, +). Alors les systèmes HVLV non libres sont non linéaire dans cette structure.
Nous montrons tout d'abord que la modélisation des systèmes HVLV à décision libre par l'algèbre (max, +) se ramène à un modèle mathématique couramment utilisé par l'automaticien : c'est une représentation d'état linéaire similaire à celle utilisée pour représenter les systèmes linéaires invariants continus [63–66]. Partant de cette représentation d'état, nous proposons un enrichissement de ce modèle par l'introduction des nouvelles variables de contrôle et de décision afin de pouvoir exhiber la dynamique non linéaire d'un système HVLV non libre. Nous discuterons par la suite la problématique de commande des systèmes libres à travers des structures de pilotage conventionnelles pour résoudre un problème de suivi de trajectoire. Cette étude a pour objectif principal de mettre l'accent sur l'insuffisance et la non adéquation de cette vision dans le cadre des systèmes non-linéaires et non libres. Pour remédier à ces problèmes, nous aborderons le pilotage des systèmes HVLV non libres sous l'angle d'une méthodologie d'ordonnancement conjuguant un objectif de suivi de trajectoire avec une optimisation non-linéaire.

2.2 Algèbre des dioïdes

Dans cette section, nous donnons quelques propriétés et définitions sur l'algèbre des dioïdes. Pour plus de détails, nous recommandons aux lecteurs les ouvrages suivants [9,39].

Définition 2.1. (**Dioïde** [39]) Un dioïde $(D, \oplus, \otimes)$ est un ensemble ordonné D muni des deux opérations, $\oplus$ et $\otimes$, qui satisfont les axiomes suivants pour tout $a, b, c \in D$:

1. l'addition $\oplus$ est associative, i.e. $(a \oplus b) \oplus c = a \oplus (b \oplus c) = a \oplus b \oplus c$;

2. l'addition est commutative, i.e. $a \oplus b = b \oplus a$;
3. l'addition est idempotente, i.e. $a \oplus a = a$;
4. la multiplication $\otimes$ est associative, i.e. $(a \otimes b) \otimes c = a \otimes (b \otimes c) = a \otimes b \otimes c$;
5. la multiplication est distributive vis-à-vis de l'addition, i.e. $(a \oplus b) \otimes c = (a \otimes c) \oplus (b \otimes c)$ et $c \otimes (a \oplus b) = (c \otimes a) \oplus (c \otimes b)$;
6. l'ensemble D inclut un élément neutre ϵ tel que $a \oplus \epsilon = a$ (ϵ est absorbant pour la multiplication, i.e. $a \otimes \epsilon = \epsilon \otimes a = \epsilon$) ;
7. l'ensemble D inclut un élément identitée tel que $a \otimes e = e \otimes a = a$.

Un dioïde est dit commutatif si la multiplication est commutative.

Quelques exemples de dioïdes :

- $(\mathbb{Z} \cup \{-\infty, +\infty\}, max, +)$ est un dioïde commutatif avec : $\oplus$ est le max et $\otimes$ est le $+$. Il est noté $\bar{\mathbb{Z}}_{max}$. Dans ce dioïde, $\epsilon = -\infty$ et $e = 0$. Il est dénommé *algèbre (max,+)*.
- $(\mathbb{Z} \cup \{-\infty, +\infty\}, min, +)$ est un dioïde commutatif avec : $\oplus$ est le min et $\otimes$ est le $+$. Il est noté $\bar{\mathbb{Z}}_{min}$. Ce dioïde est isomorphe au dioïde *(min, +)* et il est appelé *algèbre (min,+)*.

2.3 Modélisation des systèmes HVLV en utilisant l'algèbre (max,+)

A notre connaissance, la plupart des travaux de recherche sur l'algèbre (max,+) ont abordé la représentation des systèmes de production, y compris les systèmes HVLV à décision libre. Nous rappelons dans cette section les modèles (max,+) pour ces derniers. Nous proposons par la suite une représentation dans l'algèbre (max,+) des systèmes HVLV à décision non libre.

2.3.1 Modèle HVLV libre

Il a été démontré dans la littérature que les systèmes libres (voir figure 2.1) peuvent être décrits par des équations linéaires dans l'algèbre (max,+) [9, 36]. Ces équations peuvent être mises sous la forme d'état suivante :

$$\begin{cases} X(T+1) = A \otimes X(T) \oplus B \otimes U(T+1) \\ Y(T+1) = C \otimes X(T+1) \end{cases} \tag{2.1}$$

avec :

- $X(T)$ est le vecteur d'état du système pendant la période T (T est le nombre d'occurrence de l'état X). Il est de dimension $(n \times 1)$. Il contient les dates de début des opérations des produits sur les machines pendant la période T (n est le nombre total des opérations),

- $U(T)$ est le vecteur d'entrée (vecteur de contrôle) pendant la période T. Il est de dimension $(l \times 1)$. Il collecte les dates d'entrée de la matière première des produits dans le système pendant la période T (l est le nombre total de produits),
- $Y(T)$ est le vecteur de sortie du système pendant la période T. Dans ce mémoire, Y est de dimension $(n \times 1)$. Il contient les dates de fin des opérations pendant la période T.
- A est une matrice carrée appropriée de dimension $(n \times n)$ à éléments dans l'algèbre (max,+).
- B est une matrice appropriée de dimension $(n \times 1)$ à éléments dans l'algèbre (max,+).
- C est une matrice (max, +) carrée adéquate de dimension $(n \times n)$ qui contient les temps opératoires des opérations.

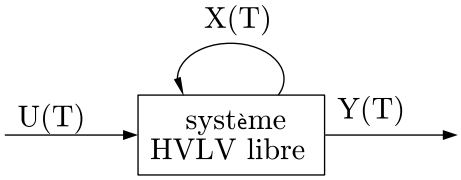

Figure 2.1 – Modèle HVLV libre

La classe des systèmes HVLV à décision libre que nous étudions ici est celle qui met en jeu des phénomènes de synchronisation qui nécessitent une disponibilité de toutes les ressources en même temps. Cette catégorie des systèmes exclut tout phénomène de concurrence. La structure linéaire de l'équation (2.1) ne modélise pas les phénomènes de conflit entre les opérations sur les machines. En effet, la commande $U(T)$ dans l'équation (2.1) permet juste de résoudre un problème de poursuite de trajectoire. Ses valeurs déterminent la date de début au plus tard des opérations sur les machines afin de satisfaire les dates de besoin de produits (production en juste-à-temps) tout en supposant que l'ordre des produits concurrents sur les mêmes machines est connu à l'avance.

D'une manière générale, un système HVLV est vu comme un réseau de machines et de stocks. Considérons un système HVLV composé d'un ensemble de m machines $\{M_k\}$, $(k = 1, \ldots m)$ dans lequel nous fabriquons un ensemble de l produits (*jobs*) $\{J_i\}$, $(i = 1, \ldots l)$. Chaque produit est composé de r opérations.
Afin de construire le modèle 2.1, les hypothèses ci-dessous sont considérées :

- les temps de transfert des pièces entre les différentes machines sont nuls et ne sont pas pris en considération dans le modèle,
- nous supposons qu'il n'y a pas de changement de série entre les différents types,
- l'ordre des différents types de produits sur les machines est connu à l'avance,
- le cas des pannes et des réparations des machines n'est pas considéré,
- la capacité de chaque machine est égale à 1 (l'ouverture est continue),

Considérons maintenant les notations suivantes qui seront utilisées dans toute la suite de ce document :

- O_{ijk} $(1 \leq i \leq l, 1 \leq j \leq n, 1 \leq k \leq m)$ est l'opération j du job $\{J_i\}$ traitée sur la machine M_k.
- $x_{ijk}(T)$ est la date de début de l'opération O_{ijk} sur la période T.
- $y_{ijk}(T)$ est la date de fin de O_{ijk} sur la période T. Dans ce cas, $y_{ijk}(T)$ est égale à la somme de son temps opératoire p_{ijk} et $x_{ijk}(T)$, i.e., $y_{ijk}(T) = x_{ijk}(T) + p_{ijk}$.
- $u_i(T)$ est la date d'entrée de la matière première du produit de type P_i dans le système pour la $T^{\text{ème}}$ fois.

Sur la période $T+1$, pour qu'une opération O_{ijk} commence sur une machine M_k, trois conditions doivent être satisfaites et le *maximum* de ces conditions fournit la date de début de cette opération. Ces conditions sont :

1. tous les prédécesseurs de O_{ijk} doivent être complétés sur M_k sur la période considérée,
2. la machine concernée doit être disponible pour le traitement de O_{ijk} pendant la période $T+1$, c-à-d la machine doit être libérée de tous les produits exécutés pendant la période T,
3. les dates d'entrée de produit $u_i(T+1)$ doivent être respectées.

Dans ce contexte et en se basant sur les gammes des différents types de produits, deux situations sont distinguées :

- **cas 1 :** si O_{ijk} est la première opération dans son job J_i, alors sa date de début $x_{ijk}(T+1)$ est déterminée par le *maximum* de l'ensemble $(x_{i'j'k}(T)+p_{i'j'k}, x_{efk}(T+1)+p_{efk}, u_i(T+1))$ avec :
  - $x_{i'j'k}(T)$ est la date de début de la dernière opération $O_{i'j'k}$ traitée sur la machine M_k sur la période T (i, i' et j, j' peuvent être égaux, tout dépend de l'ordre des opérations fixé dans la période T).
  - $x_{efk}(T+1)$ est la date de début de toutes les opérations O_{efk} qui précèdent O_{ijk} sur M_k sur la période $T+1$, vu que l'ordre des opérations sur les machines est supposé connu à l'avance ($e \neq i$).

  Une expression de cette situation peut être formalisée mathématiquement comme suit :

$$\begin{cases} x_{ijk}(T+1) = max(x_{i'j'k}(T) + p_{i'j'k}; x_{efk}(T+1) + p_{efk}; u_i(T+1)) \\ y_{ijk}(T+1) = x_{ijk}(T+1) + p_{ijk} \end{cases} \tag{2.2}$$

  En utilisant les notations de l'algèbre (max, +), l'équation (2.2) devient :

$$\begin{cases} x_{ijk}(T+1) = x_{i'j'k}(T) \otimes p_{i'j'k} \oplus x_{efk}(T+1) \otimes p_{efk} \oplus u_i(T+1) \\ y_{ijk}(T+1) = x_{ijk}(T+1) \otimes p_{ijk} \end{cases} \tag{2.3}$$

- **cas 2 :** si O_{ijk} n'est pas la première opération dans son job (produit) J_i (elle a des prédécesseurs), alors sa date de début $x_{ijk}(T+1)$ est déterminée par le *maximum* de l'ensemble $(x_{i(j-1)t}(T+1) + p_{i(j-1)t}, x_{efk}(T+1) + p_{efk})$ avec $x_{i(j-1)l}(T+1)$ est

la date de début du prédécesseur direct $O_{i(j-1)t}$ de l'opération O_{ijk} traitée sur la machine M_t pendant la période $T+1$. Une expression de cette situation peut être formalisée mathématiquement comme suit :

$$\begin{cases} x_{ijk}(T+1) = max(x_{i(j-1)t}(T+1) + p_{i(j-1)t}; x_{efk}(T+1) + p_{efk}) \\ y_{ijk}(T+1) = x_{ijk}(T+1) + p_{ijk} \end{cases} \tag{2.4}$$

L'équation (2.4) peut être réécrite dans l'algèbre (max, +) de la manière suivante :

$$\begin{cases} x_{ijk}(T+1) = x_{i(j-1)t}(T+1) \otimes p_{i(j-1)t} \oplus x_{efk}(T+1) \otimes p_{efk} \\ y_{ijk}(T+1) = x_{ijk}(T+1) \otimes p_{ijk} \end{cases} \tag{2.5}$$

Exemple illustratif : Pour des raisons de simplicité et sans perte de généralité, nous donnons ici un exemple illustratif qui explique la construction du modèle (2.1). La méthodologie de modélisation proposée est générique et reste applicable à n'importe quel système de type HVLV.
Nous considérons dans cet exemple un système HVLV libre de type flow-shop généralisé. Il est composé de 2 machines (M_1 et M_2). Dans ce système 2 types de produits sont fabriqués (P_1 et P_2). Nous supposons que P_1 est traité avant P_2 sur M_2. Les routes suivies par les différents types de produits sont représentées par la figure 2.2. Dans l'exemple représenté

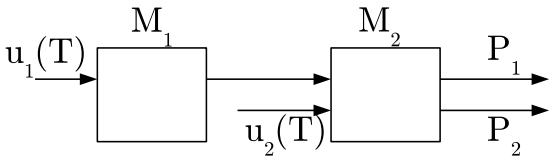

Figure 2.2 – Système HVLV avec deux types de produits

sur la figure 2.2, nous avons :

- $l = 2$ (deux types de produits),
- $n = 3$ (trois opérations),
- $m = 2$ (deux machines).

En se basant sur la méthodologie de modélisation proposée ci-dessus, nous avons la dynamique du système suivante :

$$\begin{cases} x_{111}(T+1) = max(x_{111}(T) + p_{111}; u_1(T+1)) \\ x_{122}(T+1) = max(x_{212}(T) + p_{212}; x_{111}(T+1) + p_{111}) \\ x_{212}(T+1) = max(x_{122}(T+1) + p_{122}; u_2(T+1)) \end{cases}$$

$$\begin{cases} y_{111}(T+1) = x_{111}(T+1) + p_{111} \\ y_{122}(T+1) = x_{122}(T+1) + p_{122} \\ y_{212}(T+1) = x_{212}(T+1) + p_{212} \end{cases}$$

Les équations ci dessus peuvent être réécrites dans l'algèbre (max, +) comme suit :

$$\begin{cases} x_{111}(T+1) = x_{111}(T) \otimes p_{111} \oplus u_1(T+1) \\ x_{122}(T+1) = x_{212}(T) \otimes p_{212} \oplus x_{111}(T+1) \otimes p_{111} \\ x_{212}(T+1) = x_{122}(T+1) \otimes p_{122} \oplus u_2(T+1) \end{cases}$$

En remplaçant $x_{111}(T+1)$ par sa valeur dans l'expression de $x_{122}(T+1)$ et $x_{122}(T+1)$ par sa valeur dans l'expression de $x_{212}(T+1)$ et en utilisant la distributivité de $\otimes$ par rapport à $\oplus$, nous obtenons :

$$x_{111}(T+1) = x_{111}(T) \otimes p_{111} \oplus u_1(T+1)$$

$$x_{122}(T+1) = x_{111}(T) \otimes p_{111} \otimes p_{111} \oplus x_{212}(T) \otimes p_{212} \oplus u_1(T+1) \otimes p_{111}$$

$$x_{212}(T+1) = x_{111}(T) \otimes p_{111} \otimes p_{111} \otimes p_{122} \oplus u_1(T+1) \otimes p_{111} \otimes p_{122} \oplus u_2(T+1)$$

$$y_{111}(T+1) = x_{111}(T+1) \otimes p_{111}$$

$$y_{122}(T+1) = x_{122}(T+1) \otimes p_{122}$$

$$y_{212}(T+1) = x_{212}(T+1) \otimes p_{212}$$

Les équations ci dessus peuvent être mises sous la forme d'état (2.1), avec :

$$X(T+1) = \begin{bmatrix} x_{111}(T+1) \\ x_{122}(T+1) \\ x_{212}(T+1) \end{bmatrix}$$

$$U(T+1) = \begin{bmatrix} u_1(T+1) \\ u_2(T+1) \end{bmatrix}$$

$$Y(T+1) = \begin{bmatrix} y_{111}(T+1) \\ y_{122}(T+1) \\ y_{212}(T+1) \end{bmatrix}$$

$$A = \begin{bmatrix} p_{111} & \epsilon & \epsilon \\ p_{111} \otimes p_{111} & \epsilon & p_{212} \\ p_{111} \otimes p_{111} \otimes p_{122} & \epsilon & \epsilon \end{bmatrix}$$

$$B = \begin{bmatrix} 0 & \epsilon \\ p_{111} & \epsilon \\ p_{111} \otimes p_{122} & 0 \end{bmatrix}$$

$$C = \begin{bmatrix} p_{111} & \epsilon & \epsilon \\ \epsilon & p_{122} & \epsilon \\ \epsilon & \epsilon & p_{212} \end{bmatrix}$$

Vu que le problème d'ordonnancement est supposé résolu en amont, la commande $U(T)$ n'est pas capable de générer des ordonnancements sur les machines (équation (2.1)). Dans ce cas, une modélisation linéaire des systèmes HVLV au sens de l'algèbre (max,+) n'est pas adéquate à notre problématique de recherche où nous devons traiter des problèmes de concurrence. Afin de remédier à ce problème et de décrire le comportement complet du système, la méthodologie de modélisation doit tenir compte des phénomènes de conflits sur les machines. En effet, une stratégie de prise de décision doit être introduite dans le modèle. Pour ce faire, une nouvelle composante de commande doit être intégrée dans le modèle afin de résoudre le conflit entre les opérations concurrentes sur les mêmes machines. Cette commande sera introduite à l'aide des variables de décision (variables de contrôle). Selon les valeurs choisies pour ces variables, un ordonnancement faisable sera généré. Afin d'éviter le chevauchement des opérations, des contraintes doivent être également introduites dans le modèle.

2.3.2 Modèle HVLV non libre

Modèle HVLV non libre sans maintenance

Comme indiqué dans la section 2.3.1 (p.28), les systèmes HVLV libres peuvent être décrits par une forme d'état linéaire dans l'algèbre (max,+). Cette linéarité ne permet pas de résoudre des problèmes de concurrence sur les machines dans les systèmes non libre. L'introduction d'une stratégie de prise de décision dans le modèle (2.1), qui consiste à décider l'ordre entre les produits sur les machines, semble incontournable afin de résoudre le problème de conflit.

En effet, cette stratégie va nous permettre de décider de l'ordre des opérations concurrentes sur les machines. Dans [9] (chapitres 1 et 9), les auteurs indiquent qu'une non-linéarité dans le modèle (max,+) est nécessaire pour ce type de problème. Par conséquent, le modèle (2.1) sera augmenté par l'intégration d'une nouvelle composante de commande (variables de décision). La dynamique du système non libre (voir figure 2.3) est décrite par l'équation d'état non linéaire suivante [67] :

$$\begin{cases} X(T+1) = A \otimes X(T) \oplus B \otimes U(T+1) \oplus V(T+1) \otimes X(T+1) \\ Y(T+1) = C \otimes X(T+1) \end{cases} \tag{2.6}$$

où : $X(T)$, $U(T)$, $Y(T)$, A, B, C ont la même signification que dans le modèle (2.1) et

$V(T)$ est une matrice (max,+) de décision (contrôle) pendant la période T. Elle est de dimension $(n \times n)$. $V(T)$ est composée de toutes les variables de décision pendant la période T, les temps opératoires et l'élément neutre pour l'addition ϵ (p.62-63).

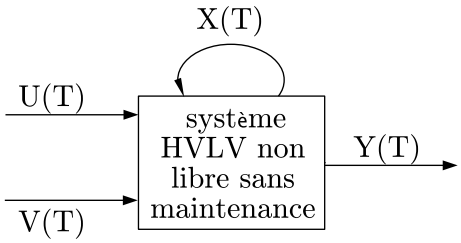

Figure 2.3 – Modèle HVLV non libre sans maintenance

Les mêmes hypothèses de travail données dans la section 2.3.1 sont de nouveau considérées ici pour élaborer le modèle non libre.
Pour qu'une opération O_{ijk} commence sur une machine M_k, trois conditions doivent être satisfaites et le *maximum* de ces conditions fournit la date de début de cette opération. Ces conditions sont :

1. tous les prédécesseurs de O_{ijk} doivent être complétés pendant la période $T+1$: les contraintes des gammes sont satisfaites et les opérations $O_{i''j''k}(i'' \neq i$ et $j'' \neq j)$ nécessitant un traitement sur une même machine M_k pendant la période $T+1$ sont traitées. L'ordre entre O_{ijk} et $O_{i"j"k}$ sera déterminé par la valeur de la variable de décision $v_{ijk;i''j''k}(T+1)$,
2. la machine concernée doit être disponible pour le traitement de O_{ijk} pendant la période $T+1$, c-à-d la machine doit être libérée de tous les produits exécutés pendant la période T,
3. les dates d'entrée de produit $u_i(T+1)$ doivent être respectées.

Dans ce contexte et en se basant sur les gammes des différents types de produits, deux situations sont distinguées :

- **cas 1 :** si O_{ijk} est la première opération dans son job J_i, alors sa date de début $x_{ijk}(T+1)$ est déterminée par le *maximum* de l'ensemble $(x_{i'j'k}(T)+p_{i'j'k}, x_{i''j''k}(T+1)+p_{i''j''k}+v_{ijk;i''j''k}(T+1), u_i(T+1))$ avec $x_{i'j'k}(T)$ est la date de début de toutes les opérations $O_{i'j'k}$ traitées sur la machine M_k sur la période T. $x_{i''j''k}(T+1)$ est la date de début de l'opération $O_{i"j"k}$ sur la période $(T+1)$. Une expression de cette situation peut être formalisée mathématiquement comme suit :

$$\begin{cases} x_{ijk}(T+1) = max(x_{i'j'k}(T) + p_{i'j'k}; x_{i''j''k}(T+1) \\ \qquad\qquad + p_{i''j''k} + v_{ijk;i''j''k}(T+1); u_i(T+1)) \\ y_{ijk}(T+1) = x_{ijk}(T+1) + p_{ijk} \end{cases} \tag{2.7}$$

En utilisant les notations de l'algèbre (max, +), l'équation (2.7) devient :

$$\begin{cases} x_{ijk}(T+1) = x_{i'j'k}(T) \otimes p_{i'j'k} \oplus x_{i''j''k}(T+1) \otimes \\ \qquad\qquad p_{i''j''k} \otimes v_{ijk;i''j''k}(T+1) \oplus u_i(T+1) \\ y_{ijk}(T+1) = x_{ijk}(T+1) \otimes p_{ijk} \end{cases} \tag{2.8}$$

- **cas 2 :** si O_{ijk} n'est pas la première opération dans son job (produit) J_i (elle a des prédécesseurs), alors sa date de début $x_{ijk}(T+1)$ est déterminée par le *maximum* de l'ensemble $(x_{i'j'k}(T) + p_{i'j'k}, x_{i(j-1)t}(T+1) + p_{i(j-1)t}, x_{i''j''k}(T+1) + p_{i''j''k} + v_{ijk;i''j''k}(T+1))$ avec $x_{i(j-1)t}(T+1)$ est la date de début du prédécesseur direct $O_{i(j-1)t}$ de l'opération O_{ijk} traité sur la machine M_t sur la période $T+1$. Une expression de cette situation peut être formalisée mathématiquement comme suit :

$$\begin{cases} x_{ijk}(T+1) = max(x_{i'j'k}(T) + p_{i'j'k}; x_{i(j-1)t}(T+1) + p_{i(j-1)t}; \\ \qquad\qquad x_{i''j''k}(T+1) + p_{i''j''k} + v_{ijk;i''j''k}(T+1)) \\ y_{ijk}(T+1) = x_{ijk}(T+1) + p_{ijk} \end{cases} \tag{2.9}$$

L'équation (2.9) peut être réécrite dans l'algèbre (max, +) de la façon suivante :

$$\begin{cases} x_{ijk}(T+1) = x_{i'j'k}(T) \otimes p_{i'j'k} \oplus x_{i(j-1)t}(T+1) \\ \qquad\qquad \otimes p_{i(j-1)t} \oplus x_{i''j''k}(T+1) \\ \qquad\qquad \otimes p_{i''j''k} \otimes v_{ijk;i''j''k}(T+1) \\ y_{ijk}(T+1) = x_{ijk}(T+1) \otimes p_{ijk} \end{cases} \tag{2.10}$$

D'après les équations (2.8) et (2.10), nous remarquons une multiplication dans l'algèbre (max,+) entre les variables d'état du système $x_{ijk}(T+1)$ et les variables de contrôle (de décision) $v_{ijk;i''j''k}(T+1)$. Ceci entraîne une non-linéarité dans le modèle (max,+). La complexité engendrée par cette non-linéarité consiste dans le manque de techniques permettant de résoudre des modèles non-linéaires dans l'algèbre des dioïdes, (contrairement aux modèles libres linéaires où nous disposons des moyens qui facilitent leur résolution : La *résiduation* est l'une de ces moyens qui est utilisée pour inverser les systèmes (max,+)). En plus, la taille des modèles (max,+) non-linéaires est beaucoup plus grande que celle des modèles linéaires due à l'intégration d'autres variables de contrôle (les variables de décision). Ceci augmente la complexité de résolution de ce genre de modèles.

Exemple illustratif : Nous illustrons à travers l'exemple précédent représenté par la figure 2.2 la construction du modèle 2.6.
En se basant sur la méthodologie de modélisation proposée ci-dessus, nous avons la dynamique du système non libre suivante :

$$x_{111}(T+1) = max(x_{111}(T) + p_{111}; u_1(T+1))$$

$$x_{122}(T+1) = max(x_{122}(T)+p_{122}; x_{212}(T)+p_{212}; x_{111}(T+1)+p_{111}; x_{212}(T+1)+ p_{212}+v_{122;212}(T+1))$$

$$x_{212}(T+1) = max(x_{122}(T)+p_{122}; x_{212}(T)+p_{212}; x_{122}(T+1)+p_{122}+v_{212;122}(T+1); u_2(T+1))$$

$$y_{111}(T+1) = x_{111}(T+1)+p_{111}$$

$$y_{122}(T+1) = x_{122}(T+1)+p_{122}$$

$$y_{212}(T+1) = x_{212}(T+1)+p_{212}$$

Les équations ci dessus peuvent être réécrites dans l'algèbre (max, +) comme suit :

$$x_{111}(T+1) = x_{111}(T) \otimes p_{111} \oplus u_1(T+1)$$

$$x_{122}(T+1) = x_{122}(T) \otimes p_{122} \oplus x_{212}(T) \otimes p_{212} \oplus x_{111}(T+1) \otimes p_{111} \oplus x_{212}(T+1) \otimes p_{212} \otimes v_{122;212}(T+1)$$

$$x_{212}(T+1) = x_{122}(T) \otimes p_{122} \oplus x_{212}(T) \otimes p_{212} \oplus x_{122}(T+1) \otimes p_{122} \otimes v_{212;122}(T+1) \oplus u_2(T+1)$$

En remplaçant $x_{111}(T+1)$ par sa valeur dans l'expression de $x_{122}(T+1)$ et en utilisant la distributivité de $\otimes$ par rapport à $\oplus$, nous obtenons :

$$x_{111}(T+1) = x_{111}(T) \otimes p_{111} \oplus u_1(T+1)$$

$$x_{122}(T+1) = x_{111}(T) \otimes p_{111} \otimes p_{111} \oplus x_{122}(T) \otimes p_{122} \oplus x_{212}(T) \otimes p_{212} \oplus u_1(T+1) \otimes p_{111} \oplus x_{212}(T+1) \otimes p_{212} \otimes v_{122;212}(T+1)$$

$$x_{212}(T+1) = x_{122}(T) \otimes p_{122} \oplus x_{212}(T) \otimes p_{212} \oplus x_{122}(T+1) \otimes p_{122} \otimes v_{212;122}(T+1) \oplus u_2(T+1)$$

$$y_{111}(T+1) = x_{111}(T+1) \otimes p_{111}$$

$$y_{122}(T+1) = x_{122}(T+1) \otimes p_{122}$$

$$y_{212}(T+1) = x_{212}(T+1) \otimes p_{212}$$

Les équations ci dessus peuvent être réécrites sous la forme d'état (2.6), avec :

$$X(T+1) = \begin{bmatrix} x_{111}(T+1) \\ x_{122}(T+1) \\ x_{212}(T+1) \end{bmatrix}$$

$$U(T+1) = \begin{bmatrix} u_1(T+1) \\ u_2(T+1) \end{bmatrix}$$

$$Y(T+1) = \begin{bmatrix} y_{111}(T+1) \\ y_{122}(T+1) \\ y_{212}(T+1) \end{bmatrix}$$

$$A = \begin{bmatrix} p_{111} & \epsilon & \epsilon \\ p_{111} \otimes p_{111} & p_{122} & p_{212} \\ \epsilon & p_{122} & p_{212} \end{bmatrix}$$

$$B = \begin{bmatrix} 0 & \epsilon \\ p_{111} & \epsilon \\ \epsilon & 0 \end{bmatrix}$$

$$C = \begin{bmatrix} p_{111} & \epsilon & \epsilon \\ \epsilon & p_{122} & \epsilon \\ \epsilon & \epsilon & p_{212} \end{bmatrix}$$

$$V(T+1) = \begin{bmatrix} \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & p_{212} \otimes v_{122;212}(T+1) \\ \epsilon & p_{122} \otimes v_{212;122}(T+1) & \epsilon \end{bmatrix}$$

Modèle HVLV non libre avec maintenance

Dans cette section nous conservons les mêmes notations pour les opérations O_{ijk} de chaque job J_i. Nous supposons qu'on a $h(h = 1, \ldots, x)$ activités de maintenance périodiques dans chaque cycle de production T, notées PM_{hk} et qui sont exécutées sur chaque machine M_k selon une certaine politique de maintenance. Chaque maintenance préventive PM_{hk} possède une durée déterministe t_{hk}. Supposons que $x_{hk}(T)$ est la date de début de chaque intervention de maintenance PM_{hk} sur la période T. Donc, sa date de fin notée $y_{hk}(T)$ est égale à la somme de $x_{hk}(T)$ et t_{hk}.

Nous nous limiterons aux systèmes ayant les propriétés suivantes pour le développement du modèle non libre avec maintenance :

- les machines sur lesquelles les opérations et les tâches de maintenance doivent être exécutées sont connues,
- les durées allouées aux différentes activités de maintenance sont données et peuvent être différentes,
- les valeurs des périodes entre les opérations de maintenance sont connues,
- l'ordre des opérations de maintenance sur les machines n'est pas connu à l'avance. Dans la littérature, la plupart des travaux considèrent un ordre figé des opérations de maintenance [68, 69].

Deux types de maintenance préventive sont considérés : la maintenance répétitive périodique et la maintenance flexible périodique. Afin de spécifier la nature de maintenance, le paramètre $T_{(.)}$ est introduit. Par conséquent :

- Si $T_{(.)} = T_k$, $k = 1, \ldots, x$, alors la maintenance est dite répétitive, i.e. les intervalles de temps entre deux interventions de maintenance successives sont égaux (voir figure 2.4).

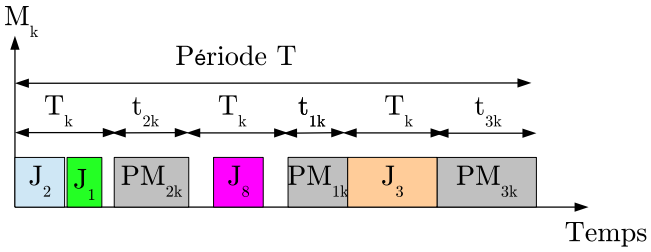

Figure 2.4 – Un ordonnancement sur une seule machine avec maintenance répétitive périodique : PM_{hk} est la $h^{ème}$ opération de maintenance ($h = 1, \ldots, x$) sur la machine M_k [3].

- Si $T_{(.)} = T_{ik,jk}$, $i, j = 1, \ldots, x$, $i \neq j$, alors la maintenance est dite flexible, i.e. les intervalles de temps entre deux interventions de maintenance successives peuvent être différents mais sont fixés à l'avance (voir figure 2.5).

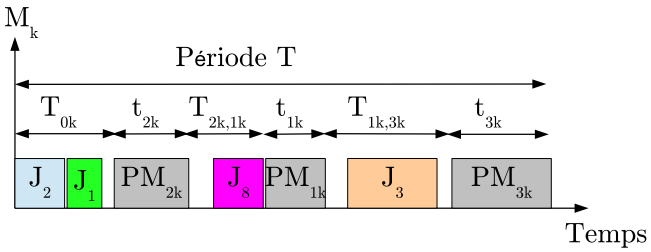

Figure 2.5 – Un ordonnancement sur une seule machine avec maintenance flexible périodique : PM_{hk} est la $h^{ème}$ opération de maintenance ($h = 1, \ldots, x$) sur la machine M_k [4].

La dynamique du système non libre avec maintenance est décrite par l'équation d'état suivante :

$$\begin{cases} X(T+1) = A \otimes X(T) \oplus B \otimes U(T+1) \oplus V(T+1) \otimes X(T+1) \\ Y(T+1) = C \otimes X(T+1) \end{cases} \tag{2.11}$$

où : $X(T)$, $U(T)$, $Y(T)$, A, B, C et V ont la même signification que dans le modèle (2.6). Concernant les dimensions des différentes matrices, le nombre n dans le modèle (2.6) est remplacé par le nombre $N = n + P$ (P est le nombre total des activités de maintenance dans le système).

Dynamique de la maintenance

En adoptant la même méthodologie de modélisation détaillée dans la section 2.3.2, deux nouvelles variables de décision notées $v_{ijk;hk}(T)$ et $v_{hk;zk}(T)$ sont introduites dans le modèle de la maintenance proposé. Les activités de maintenance sont considérées comme des opérations élémentaires.

En effet, d'une part, l'ordre entre les tâches de maintenance PM_{hk} et les opérations O_{ijk} pendant la période T sera déterminé par $v_{ijk;hk}(T)$. D'autre part, le conflit entre deux interventions de maintenance successives PM_{hk} et PM_{zk} $(h \neq z)$ pendant la période T sera résolu par $v_{hk;zk}(T)$.
Pour qu'une opération de maintenance PM_{hk} commence sur une machine M_k, deux conditions doivent être satisfaites et le *maximum* de ces conditions fournit la date de début de cette opération. Ces conditions sont :

1. tous les prédécesseurs de PM_{hk}, O_{ijk} et PM_{zk} $(z \neq h)$, nécessitant un traitement sur une même machine M_k pendant la période $T+1$ doivent être complétés,
2. la machine concernée doit être disponible pour le traitement de PM_{hk} pendant la période $T+1$, c-à-d la machine doit être libérée de tous les produits et les tâches de maintenance exécutés pendant la période T.

Par conséquent, la dynamique des activités de maintenance PM_{hk} sur un cycle de production $T+1$ peut être décrite par l'équation (max,+) suivante :

$$\begin{cases} x_{hk}(T+1) = max(x_{ijk}(T) + p_{ijk}; x_{zk}(T) + t_{zk}; x_{ijk}(T+1) \\ \qquad\qquad + p_{ijk} + v_{hk;ijk}(T+1); x_{zk}(T+1) \\ \qquad\qquad + T_{(.)} + v_{hk;zk}(T+1)) \\ y_{hk}(T+1) = x_{hk}(T+1) + t_{hk} \end{cases} \tag{2.12}$$

L'équation (2.12) peut être exprimée dans l'algèbre (max,+) comme suit :

$$\begin{cases} x_{hk}(T+1) = x_{ijk}(T) \otimes p_{ijk} \oplus x_{zk}(T) \otimes t_{zk} \\ \qquad\qquad \oplus x_{ijk}(T+1) \otimes p_{ijk} \otimes v_{hk;ijk}(T+1) \\ \qquad\qquad \oplus x_{zk}(T+1) \otimes T_{(.)} \otimes v_{hk;zk}(T+1) \\ y_{hk}(T+1) = x_{hk}(T+1) \otimes t_{hk} \end{cases} \tag{2.13}$$

Modèle non libre global avec maintenance

Afin d'établir le modèle d'ordonnancement (max,+) global avec maintenance, le modèle représenté par les équations (2.7) et (2.9) et le sous-modèle de la maintenance représenté par l'équation 2.12 sont fusionnés.
Dans le modèle global, la dynamique des opérations O_{ijk} décrite par leurs dates de début $x_{ijk}(T+1)$ est obtenue par l'augmentation du modèle non libre sans maintenance.
Afin de considérer le cas de la maintenance, l'expression de $x_{ijk}(T+1)$ dans les équations (2.7) et (2.7) est augmentée par le terme« $x_{hk}(T+1) + t_{hk} + v_{ijk;hk}(T+1)$ ». Ce terme

représente l'ordre entre O_{ijk} et PM_{hk}. Cette relation est déterminée par la variable de décision $v_{ijk;hk}(T+1)$. Dans ce cadre et en se basant sur les gammes de production de chaque job J_i, deux situations sont distinguées :

- **cas 1 :** si O_{ijk} est la première opération dans son job J_i pendant la période $(T+1)$, alors sa date de début $x_{ijk}(T+1)$ est déterminée par l'expression suivante :

$$\begin{cases} x_{ijk}(T+1) = max(x_{i'j'k}(T) + p_{i'j'k}; x_{i''j''k}(T+1) + p_{i''j''k} \\ \qquad\qquad + v_{ijk;i''j''k}(T+1); u_i(T+1); \\ \qquad\qquad x_{hk}(T+1) + t_{hk} + v_{ijk;hk}(T+1)) \\ y_{ijk}(T+1) = x_{ijk}(T+1) + p_{ijk} \end{cases} \tag{2.14}$$

  En utilisant les notations de l'algèbre (max, +), l'équation (2.14) devient :

$$\begin{cases} x_{ijk}(T+1) = x_{i'j'k}(T) \otimes p_{i'j'k} \oplus x_{i''j''k}(T+1) \otimes p_{i''j''k} \otimes v_{ijk;i''j''k}(T+1) \\ \qquad\qquad \oplus u_i(T+1) \oplus x_{hk}(T+1) \otimes t_{hk} \otimes v_{ijk;hk}(T+1) \\ y_{ijk}(T+1) = x_{ijk}(T+1) \otimes p_{ijk} \end{cases} \tag{2.15}$$

- **cas 2 :** si O_{ijk} n'est pas la première opération dans son job pendant la période $(T+1)$ (elle a des prédécesseurs), alors $x_{ijk}(T+1)$ est déterminée par l'expression suivante :

$$\begin{cases} x_{ijk}(T+1) = max(x_{i'j'k}(T) + p_{i'j'k}; x_{i(j-1)t}(T+1) + p_{i(j-1)t}; \\ \qquad\qquad x_{i''j''k}(T+1) + p_{i''j''k} + v_{ijk;i''j''k}(T+1); \\ \qquad\qquad x_{hk}(T+1) + t_{hk} + v_{ijk;hk}(T+1)) \\ y_{ijk}(T+1) = x_{ijk}(T+1) + p_{ijk} \end{cases} \tag{2.16}$$

  L'équation (2.16) peut être réécrite dans l'algèbre (max, +) comme suit :

$$\begin{cases} x_{ijk} = x_{i'j'k}(T) \otimes p_{i'j'k} \oplus x_{i(j-1)t}(T+1) \otimes p_{i(j-1)t} \\ \qquad\qquad \oplus x_{i''j''k}(T+1) \otimes p_{i''j''k} \otimes v_{ijk;i''j''k}(T+1) \\ \qquad\qquad \oplus x_{hk}(T+1) \otimes t_{hk} \otimes v_{ijk;hk}(T+1) \\ y_{ijk}(T+1) = x_{ijk}(T+1) \otimes p_{ijk} \end{cases} \tag{2.17}$$

Exemple illustratif : Nous illustrons à travers l'exemple précédent (cf. section 2.3.1) la construction du modèle (2.11). Nous supposons dans cet exemple que la machine M_2 est soumise à deux interventions de maintenance répétitives périodiques PM_{12} et PM_{22} pendant chaque cycle de production T. Soit :

- T_2 la période de la maintenance,
- x_{1k} la date de début de PM_{12} et x_{2k} la date de début de PM_{22},
- t_{1k} la durée de PM_{12} et t_{2k} la durée de PM_{22}.

Dans ce cas la dynamique du système non libre avec maintenance est donnée par :

$$x_{111}(T+1) = max(x_{111}(T) + p_{111}; u_1(T+1))$$

$$\begin{aligned} x_{122}(T+1) =& max(x_{122}(T) + p_{122}; x_{212}(T) + p_{212}; x_{12}(T) + t_{12}; x_{22}(T) + t_{22}; \\ & x_{111}(T+1) + p_{111}; x_{212}(T+1) + p_{212} + v_{122;212}(T+1); x_{12}(T+1) + t_{12} \\ & + v_{122;12}(T+1); x_{22}(T+1) + t_{22} + v_{122;22}(T+1)) \end{aligned}$$

$$\begin{aligned} x_{212}(T+1) =& max(x_{122}(T) + p_{122}; x_{212}(T) + p_{212}; x_{12}(T) + t_{12}; x_{22}(T) + t_{22}; \\ & x_{122}(T+1) + p_{122} + v_{212;122}(T+1); x_{12}(T+1) + t_{12} + v_{212;12}(T+1); \\ & x_{22}(T+1) + t_{22} + v_{212;22}(T+1); u_2(T+1)) \end{aligned}$$

$$\begin{aligned} x_{12}(T+1) =& max(x_{122}(T) + p_{122}; x_{212}(T) + p_{212}; x_{12}(T) + t_{12}; x_{22}(T) + t_{22}; x_{122}(T+1) \\ & + p_{122} + v_{12;122}(T+1); x_{212}(T+1) + p_{212} + v_{12;212}(T+1); x_{22}(T+1) \\ & + t_{22} + v_{12;22}(T+1) \end{aligned}$$

$$\begin{aligned} x_{22}(T+1) =& max(x_{122}(T) + p_{122}; x_{212}(T) + p_{212}; x_{12}(T) + t_{12}; x_{22}(T) + t_{22}; x_{122}(T+1) \\ & + p_{122} + v_{22;122}(T+1); x_{212}(T+1) + p_{212} + v_{22;212}(T+1); x_{12}(T+1) \\ & + t_{12} + v_{22;12}(T+1) \end{aligned}$$

$$y_{111}(T+1) = x_{111}(T+1) + p_{111}$$

$$y_{122}(T+1) = x_{122}(T+1) + p_{122}$$

$$y_{212}(T+1) = x_{212}(T+1) + p_{212}$$

$$y_{12}(T+1) = x_{12}(T+1) + t_{12}$$

$$y_{22}(T+1) = x_{22}(T+1) + t_{22}$$

Les équations ci dessus peuvent être réécrites dans l'algèbre (max, +) comme suit :

$$x_{111}(T+1) = x_{111}(T) \otimes p_{111} \oplus u_1(T+1)$$

$$\begin{aligned} x_{122}(T+1) =& x_{122}(T) \otimes p_{122} \oplus x_{212}(T) \otimes p_{212} \oplus x_{12}(T) \otimes t_{12} \oplus x_{22}(T) \otimes t_{22} \oplus \\ & x_{111}(T+1) \otimes p_{111} \oplus x_{212}(T+1) \otimes p_{212} \otimes v_{122;212}(T+1) \oplus x_{12}(T+1) \otimes \\ & t_{12} \otimes v_{122;12}(T+1) \oplus x_{22}(T+1) \otimes t_{22} \otimes v_{122;22}(T+1) \end{aligned}$$

$$\begin{aligned} x_{212}(T+1) =& x_{122}(T) \otimes p_{122} \oplus x_{212}(T) \otimes p_{212} \oplus x_{12}(T) \otimes t_{12} \oplus x_{22}(T) \otimes t_{22} \oplus \end{aligned}$$

$$x_{122}(T+1) \otimes p_{122} \otimes v_{212;122}(T+1) \oplus x_{12}(T+1) \otimes t_{12} \otimes v_{212;12}(T+1) \oplus$$
$$x_{22}(T+1) \otimes t_{22} \otimes v_{212;22}(T+1) \oplus u_2(T+1)$$

$$\begin{aligned} x_{12}(T+1) =& x_{122}(T) \otimes p_{122} \oplus x_{212}(T) \otimes p_{212} \oplus x_{12}(T) \otimes t_{12} \oplus x_{22}(T) \otimes t_{22} \oplus \\ & x_{122}(T+1) \otimes p_{122} \otimes v_{12;122}(T+1) \oplus x_{212}(T+1) \otimes p_{212} \otimes v_{12;212}(T+1) \oplus \\ & x_{22}(T+1) \otimes t_{22} \otimes v_{12;22}(T+1) \end{aligned}$$

$$\begin{aligned} x_{22}(T+1) =& x_{122}(T) \otimes p_{122} \oplus x_{212}(T) \otimes p_{212} \oplus x_{12}(T) \otimes t_{12} \oplus x_{22}(T) \otimes t_{22} \oplus \\ & x_{122}(T+1) \otimes p_{122} \otimes v_{22;122}(T+1) \oplus x_{212}(T+1) \otimes p_{212} \otimes v_{22;212}(T+1) \oplus \\ & x_{12}(T+1) \otimes t_{12} \otimes v_{22;12}(T+1) \end{aligned}$$

$$y_{111}(T+1) = x_{111}(T+1) \otimes p_{111}$$
$$y_{122}(T+1) = x_{122}(T+1) \otimes p_{122}$$
$$y_{212}(T+1) = x_{212}(T+1) \otimes p_{212}$$
$$y_{12}(T+1) = x_{12}(T+1) \otimes t_{12}$$
$$y_{22}(T+1) = x_{22}(T+1) \otimes t_{22}$$

En remplaçant $x_{111}(T+1)$ par sa valeur dans l'expression de $x_{122}(T+1)$ et en utilisant la distributivité de $\otimes$ par rapport à $\oplus$, nous obtenons la représentation d'état (2.11) avec :

$$X(T+1) = \begin{bmatrix} x_{111}(T+1) \\ x_{122}(T+1) \\ x_{212}(T+1) \\ x_{12}(T+1) \\ x_{22}(T+1) \end{bmatrix}$$

$$U(T+1) = \begin{bmatrix} u_1(T+1) \\ u_2(T+1) \end{bmatrix}$$

$$Y(T+1) = \begin{bmatrix} y_{111}(T+1) \\ y_{122}(T+1) \\ y_{212}(T+1) \\ y_{12}(T+1) \\ y_{12}(T+1) \end{bmatrix}$$

$$A = \begin{bmatrix} p_{111} & \epsilon & \epsilon & \epsilon & \epsilon \\ p_{111} \otimes p_{111} & p_{122} & p_{212} & t_{12} & t_{22} \\ \epsilon & p_{122} & p_{212} & t_{12} & t_{22} \\ \epsilon & p_{122} & p_{212} & t_{12} & t_{22} \\ \epsilon & p_{122} & p_{212} & t_{12} & t_{22} \end{bmatrix}$$

$$B = \begin{bmatrix} 0 & \epsilon \\ p_{111} & \epsilon \\ \epsilon & 0 \\ \epsilon & \epsilon \\ \epsilon & \epsilon \end{bmatrix}$$

$$C = \begin{bmatrix} p_{111} & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & p_{122} & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & p_{212} & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & t_{12} & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon & t_{22} \end{bmatrix}$$

La matrice de contrôle $V(T+1)$ s'écrit sous la forme suivante :

$$\begin{bmatrix} \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & p_{212} \otimes v_{122;212}(T+1) & t_{12} \otimes v_{122;12}(T+1) & t_{22} \otimes v_{122;22}(T+1) \\ \epsilon & p_{122} \otimes v_{212;122}(T+1) & \epsilon & t_{12} \otimes v_{122;12}(T+1) & t_{22} \otimes v_{122;22}(T+1) \\ \epsilon & p_{122} \otimes v_{12;122}(T+1) & p_{212} \otimes v_{12;212}(T+1) & \epsilon & t_{22} \otimes v_{12;22}(T+1) \\ \epsilon & p_{122} \otimes v_{22;122}(T+1) & p_{212} \otimes v_{22;212}(T+1) & t_{12} \otimes v_{22;12}(T+1) & \epsilon \end{bmatrix}$$

2.4 Pilotage des systèmes HVLV en utilisant l'algèbre (max,+)

2.4.1 Pilotage des systèmes HVLV libres

La commande d'un système HVLV à décision libre modélisé à l'aide du formalisme (max,+) consiste à déterminer les valeurs adéquates de ses entrées en vue d'obtenir un comportement spécifié et désiré. Plus concrètement, il s'agit ici de fixer les dates d'occurrences des événements d'entrée pour le système considéré (dates d'entrée de la matière première de produits dans le système). Depuis une vingtaine d'années, plusieurs approches complémentaires ont été proposées (commande (max, +) en boucle ouverte [9], commande par modèle interne [70], commande prédictive [60],...). Nous proposons ici un bref état de l'art sur la commande des systèmes (max, +) linéaires ainsi que son application pour le pilotage des systèmes HVLV libres.

Nous considérons ici les problèmes de poursuite de trajectoire, à savoir la date de besoin des produits. La commande appliquée vise à synthétiser une trajectoire d'entrée (commande $U(T)$) de sorte que la réponse en sortie soit la plus proche possible d'une trajectoire de référence, également appelée consigne. Celle-ci correspond à un comportement désiré en sortie du système. Lorsque le critère du juste-à-temps est considéré, l'objectif de la commande revient à calculer une trajectoire d'entrée afin que la sortie obtenue soit plus petite

ou égale à une trajectoire de référence. Autrement dit, dans le cas du juste-à-temps, le problème de poursuite de trajectoire consiste à rechercher la plus grande commande $U(T)$ telle que les dates de besoin de produits sont respectées. Cet objectif particulier induit que la sortie obtenue ne dépasse jamais la trajectoire de référence [43,71,72]. Dans ce contexte, on peut caractériser chaque commande selon deux points de vue :

- la définition d'un critère à optimiser,
- la structure adoptée pour la réalisation de la commande.

Critères

Un critère est généralement défini par une fonction, appelée fonction coût, que l'on cherche à optimiser.

- Les premiers travaux concernant le pilotage des systèmes (max, +) linéaires ont considéré le critère de juste-à-temps. La problématique du juste-à-temps est d'activer les événements d'entrée aux dates les plus tardives tout en satisfaisant certains objectifs de contrôle [73–75].
  La fonction coût que l'on cherche à maximiser quantifie donc les dates d'occurrences des événements d'entrée ce qui a pour conséquence, si on s'intéresse à un système de production, de minimiser les en-cours ou stocks internes. Dans le milieu industriel, on parle de production en flux tendus (par opposition à la production en flux poussés) qui consiste à produire la quantité juste nécessaire au moment où l'on en a besoin.
- Plus récemment, un critère différent a été considéré notamment dans [76,77]. La commande utilisée correspond à l'adaptation de la commande prédictive aux systèmes (max, +) linéaires. Le critère usuel (à minimiser) se décompose en une somme pondérée de deux sous-critères qui correspondent, d'une part, à l'erreur quadratique entre une trajectoire de référence connue et la sortie prédite du système, et d'autre part, à l'amplitude des variations de la commande (en vue de "ménager" les actionneurs). Ce dernier sous critère peut éventuellement être remplacé par l'énergie de la commande. Il est représenté par une fonction coût qui correspond à l'inverse de la somme des dates de tirs de toutes les entrées, soit :

$$J_{in} = -\sum_{j=1}^{N_p}\sum_{i=1}^{m} u_i(k+j-1),$$

  pour un système à m entrées où N_p désigne l'horizon de prédiction. Le but de ce sous critère est de maximiser la somme des dates d'activation des événements d'entrées (en ce sens, ce sous critère se rapproche de celui du juste-à-temps exposé précédemment). Un critère global considéré par [76] est donné par :

$$J = J_{out} + \lambda J_{in},$$

où J_{out} représente le critère de poursuite de trajectoire et λ est une variable scalaire utilisée pour pondérer le critère de commande.

Structure de la commande

Commande en boucle ouverte : En boucle ouverte, on applique une valeur de commande à l'entrée du système considéré et on suppose que sa sortie réagira de façon connue en fonction de cette valeur, sans contrôler le bon déroulement de cette action. Ce mode de commande ne nécessite aucune mesure des variables du système.
Concernant les systèmes (max, +) linéaires, une structure de commande en boucle ouverte a été proposée en premier lieu dans [42]. Dans cette étude, le critère considéré est le juste-à-temps et l'objectif de commande se traduit par une poursuite de trajectoire de référence. Nous rappelons ici la méthode de synthèse exploitée dans [42].
Nous avons vu dans la section 2.3.1, qu'un système HVLV à décision libre peut être décrit par le modèle d'état d'ordre n représenté par l'équation (2.1) pour tout $T \in \mathbb{N} \setminus \{0\}$ avec $X(0) = X_0$.
Nous pouvons déterminer à partir de (2.1) le comportement d'entrée-sortie d'un système. On a :

$$X(1) = A \otimes X(0) \oplus B \otimes U(1)$$

$$X(2) = A \otimes X(1) \oplus B \otimes U(2) = A^{\otimes 2} \otimes X(0) \oplus A \otimes B \otimes U(1) \oplus B \otimes U(2)$$

$$\vdots$$

$$X(T) = A^{\otimes T} \otimes X(0) \bigoplus_{i=0}^{T-1} A^{T-i-1} \otimes B \otimes U(i+1)$$

pour $T = 0, 1, 2, \ldots$
où la somme max-algébrique vide $\bigotimes_{i=0}^{-1} \ldots$ est égale par définition à $\epsilon_{n\times 1}$.
D'où,

$$Y(T) = C \otimes A^{\otimes T} \otimes X(0) \bigoplus_{i=0}^{T-1} C \otimes A^{T-i-1} \otimes B \otimes U(i+1) \tag{2.18}$$

pour $T = 0, 1, 2, \ldots$
Si on pose : $p \in \mathbb{N}\backslash\{0\}$ (p est le nombre des cycles de production), $Y = [Y(1) \quad Y(2) \quad \ldots \quad Y(p)]^T$ et $U = [U(1) \quad U(2) \quad \ldots \quad U(p)]^T$, $X(0) = \epsilon_{n\times 1}$ (c-à-d on suppose que les stocks de la matière première sont initialement vides et les machines sont initialement inoccupées). L'équation (2.18) peut s'exprimer comme suit :

$$Y = H \otimes U$$

avec

$$H = \begin{bmatrix} C \otimes B & \epsilon & \cdots & \epsilon \\ C \otimes A \otimes B & C \otimes B & \cdots & \epsilon \\ \vdots & \vdots & \ddots & \vdots \\ C \otimes A^{\otimes p-1} \otimes B & C \otimes A^{\otimes p-2} \otimes B & \cdots & C \otimes B \end{bmatrix}$$

Si on connaît la date d'entrée U de matière première dans le système, on peut calculer la date de fin de produits Y. Si on connaît le vecteur Y, on peut calculer le vecteur U en utilisant le théorème de la résiduation [9](chapitre 4).
D'une manière générale, le problème de la commande en boucle ouverte illustré sur la figure 2.6 peut être posé en les termes suivants : à partir d'un système HVLV à décision libre dont on connaît la matrice de transfert $H \in \mathbb{R}_\epsilon^{(T\times l)\times(T\times l)}$ (l est le nombre des produits). On désire, à l'aide des entrées $U \in \mathbb{R}_\epsilon^l$, faire en sorte que les sorties du système $Y \in \mathbb{R}_\epsilon^l$ suivent au mieux des trajectoires désirées dites des consignes $Z \in \mathbb{R}_\epsilon^l$. Dans [42], il est montré que ce problème a une solution optimale, c'est-à-dire qu'il existe une plus grande commande d'entrée $U_{opt} \in \mathbb{R}_\epsilon^l$ telle que la sortie résultant de cette entrée ($Y_{opt} = H \otimes U_{opt}$) soit inférieure ou égale à la sortie désirée Z. La commande U_{opt} est alors optimale vis-à-vis du critère de juste-à-temps (la sortie Y_{opt} est en juste-à-temps).

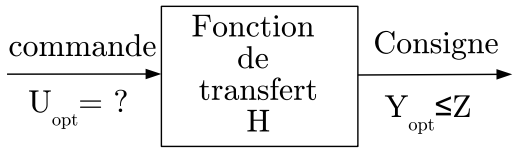

Figure 2.6 – Problématique générale de la commande en boucle ouverte

Formellement, soit $L_H : \mathbb{R}_\epsilon^l \to \mathbb{R}_\epsilon^l$, $U \to H \otimes U$, une application définie sur des dioïdes complets. Déterminer la plus grande commande revient à s'intéresser à l'ensemble

$$\{U \in \mathbb{R}_\epsilon^l \mid L_H(U) \leq Z\}$$

La commande optimale U_{opt} existe et est donnée par :

$$U_{opt} = sup\{U \in \mathbb{R}_\epsilon^l \mid L_H(U) \leqslant Z\} = H \diagdown Z$$

$$U_j = \min_i (z_i - h_{ij})$$

pour $i = 1, 2, \ldots, l$ et $j = 1, 2, \ldots, l$ (h_{ij} sont les éléments de la matrice H)

Commande en boucle fermée : Une autre structure, empruntée à l'automatique conventionnelle et utilisée pour la commande des systèmes (max ; +) linéaires est la boucle de retour ou «*feedback*». Cette structure fait apparaître dans le calcul de la loi de commande la sortie ou l'état du système.
Nous rappelons ici différentes structures de commande en boucle fermée qui ont été utilisées dont l'objectif est la poursuite de trajectoires (date de besoin de produits) en juste-à-temps.

Commande prédictive sous contraintes : La commande prédictive (CP) est une méthode relativement récente qui n'a connu un réel essor que depuis le milieu des années 80 grâce aux travaux de DW Clarke et de son équipe à *Oxford* [78–81]. Toutefois, cette technique de commande, que l'on peut rattacher à la famille de Commande Prédictive par Modèle (CPM) (*Model Predictive Control (MPC))* suscite un intérêt dans le domaine industriel depuis la fin des années 70.
Les majeurs avantages de cette structure de commande sont :

- elle est applicable aux systèmes Multi-Entrée, Multi-Sortie (MEMS), y compris les systèmes HVLV,
- elle tient compte des contraintes sur les entrées et les sorties de système,
- elle peut résoudre des problèmes de poursuite de trajectoire.

Habituellement, la CPM utilise des modèles linéaires à temps discrets. Dans la littérature, cette structure de commande a été étendue à une classe de SEDs qui peuvent être décrits par des modèles linéaires dans (max, +). Pour plus de détails, nous invitons le lecteur à consulter [60, 76, 82–85]. Dans [60, 76, 82, 83] les auteurs traitent des problèmes d'optimisation non convexes et non linéaires en utilisant la programmation quadratique séquentielle (*Sequential Quadratic Programming (SQP)*) et le problème complémentaire linéaire étendu (*Extended Linear Complementarity Problem (ELCP)*).
Les principes fondamentaux de la CPM sont :

- La prédiction : le comportement futur du système est prédit dans un certain horizon appelé N_p. La prédiction est basée sur l'état courant du système, les perturbations et la commande sont planifiées.
- L'évaluation des performances à partir d'une fonction objectif.
- L'optimisation : le contrôleur calcule la commande qui optimise la fonction objectif.
- L'action de contrôle : seul le premier élément de la séquence optimale précédente est appliqué sur le système. Tous les autres éléments peuvent être oubliés car à la période d'échantillonnage suivante, les séquences sont décalées, une nouvelle sortie est mesurée et la procédure complète est répétée. Ce procédé repose sur le principe de **l'horizon fuyant**.

Dans [86], nous avons étendu la structure CPM initialement proposée par Bart De Schutter et Van Den Boom aux systèmes HVLV à décision libre sous contraintes. Dans ce cadre, une structure CPM associée à un algorithme d'optimisation convexe sous contraintes linéaires en utilisant le principe du : «*Max-Min-Plus-Scaling (MMPS) functions*» est appliquée pour résoudre un problème de poursuite de trajectoire en juste-à-temps.

Commande par modèle interne : Dans des applications réelles, la structure de la commande en boucle ouverte ne garantit pas le maintien de la sortie du système au dessous de la référence en présence d'une erreur de modélisation. C'est le cas des systèmes HVLV en présence, par exemple, d'erreurs sur les temps de production des machines. Pour remédier à ce problème, la structure de la commande en boucle ouverte est améliorée.

La commande proposée, dont la structure est représentée par la figure 2.7 est appelée CMI (Commande par Modèle Interne) [70, 87]. La CMI est étendue aux systèmes HVLV

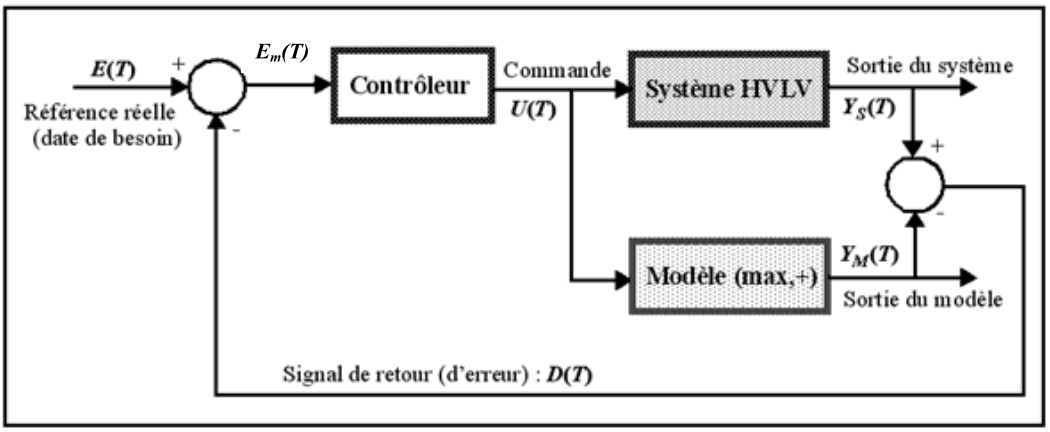

Figure 2.7 – Structure de la CMI

à décision libre [87]. Nous distinguons dans cette approche les équations qui représentent le système (2.20) de celles du modèle (2.19). Soient :

$$\begin{cases} X_M(T+1) = A_M \otimes X_M(T) \oplus B_M \otimes U(T+1) \\ Y_M(T) = C_M \otimes X_M(T) \end{cases} \tag{2.19}$$

$$\begin{cases} X_S(T+1) = A_S \otimes X_S(T) \oplus B_S \otimes U(T+1) \\ Y_S(T) = C_S \otimes X_S(T) \end{cases} \tag{2.20}$$

L'objectif de la CMI est de respecter la véracité de la relation suivante :

$$Y_S(T) \leq E(T)$$

ce qui mène, d'après la structure de la CMI, à :

$$Y_M(T) \leq Em(T)$$

car

$$E_m(T) = E(T) - Y_S(T) + Y_M(T)$$

$$Y_S(T) - E(T) = Y_M(T) - E_m(T)$$

A partir de (2.19), il est possible d'écrire :

$$Y_M(T+1) = C_M \otimes A_M \otimes X_M(T) \oplus C_M \otimes B_M \otimes U(T+1)$$

Pour obtenir la relation suivante :

$$Y_M(T+1) \leq Em(T+1)$$

il faut que :

$$C_M \otimes A_M \otimes X_M(T) \oplus C_M \otimes B_M \otimes U(T+1) \leq Em(T+1)$$

En supposant que $C_M \otimes A_M \otimes X_M(T) \leq Em(T+1)$, on a la condition suivante à satisfaire :

$$C_M \otimes B_M \otimes U(T+1) \leq Em(T+1)$$

Cette dernière expression implique que :

$$U(T+1) = C_M \otimes B_M \backslash Em(T+1)$$

avec :

$$Em(T+1) = E(T+1) - D(T+1)$$

et

$$D(T+1) = Y_S(T+1) - Y_M(T+1)$$

illustration : Nous considérons ici l'exemple utilisé dans la section 2.3.1 à savoir,

- $Y_S(T) = [y_{111}(T)\ y_{122}(T)\ y_{212}(T)]^t$: le vecteur de sortie du système,
- $Y_M(T) = [y_{M111}(T)\ y_{M122}(T)\ y_{M212}(T)]^t$: le vecteur de sortie du modèle,
- $X_S(T) = [x_{111}(T)\ x_{122}(T)\ x_{212}(T)]^t$: le vecteur d'état du système,
- $X_M(T) = [x_{M111}(T)\ x_{M122}(T)\ x_{M212}(T)]^t$: le vecteur d'état du modèle,
- p_S : les temps opératoires des opérations du système,
- p_M : les temps opératoires des opérations du modèle,
- $D(T) = [d_1(T)\ d_2(T)]^t$: le vecteur d'erreur de modélisation (*feedback*),
- $E(T) = [e_1(T)\ e_2(T)]^t$: le vecteur de référence,
- $Em(T) = [em_1(T)\ em_2(T)]^t$: le vecteur de référence modifiée.

Dans le cas industriel, une erreur de modélisation sur les paramètres du système, les temps opératoires des opérations par exemple (voir tableau 2.1), rend une structure de commande en boucle ouverte non valable. En effet la sortie réelle du système qui dépend de ces paramètres sera différente de la sortie du modèle. Par conséquent, une stratégie de commande permettant de compenser cette erreur de modélisation est indispensable. La structure de la CMI permet de compenser cette erreur.

Tableau 2.1 – Erreurs de modélisation

T	1 :4	5 :10
p_{S122}	2	2
p_{S212}	1	1
p_{M122}	1.8	2.1
p_{M212}	1.2	1.1
$d_1(T)$	0.2	-0.1
$d_1(T)$	-0.2	-0.1

avec :

$$d_1(T) = p_{S122}(T) - p_{M122}(T)$$

$$d_2(T) = p_{S212}(T) - p_{M212}(T)$$

Par convention, nous supposons que $X_M(0) = [\epsilon\ \epsilon\ \epsilon]^t$ et $X_S(0) = [\epsilon\ \epsilon\ \epsilon]^t$. Appliquons la CMI sur une période de simulation [1,10] avec les références (consignes) $e_1(T)$ et $e_2(T)$ (tableau 2.2). Nous avons utilisé la bibliothèque *Maxplus* du logiciel *ScicosLab* pour faire le calcul (max, +).

Tableau 2.2 – Résultats de simulation

T	1	2	3	4	5	6	7	8	9	10
e_1 : date de besoin de P_1	12	24	36	48	60	72	84	96	108	120
e_2 : date de besoin de P_2	13	26	39	52	65	78	91	104	117	130
d_1 : erreur de modélisation sur p_{122}	0.2	0.2	0.2	0.2	-0.1	-0.1	-0.1	-0.1	-0.1	-0.1
d_2 : erreur de modélisation sur p_{212}	-0.2	-0.2	-0.2	-0.2	-0.1	-0.1	-0.1	-0.1	-0.1	-0.1
u_1 : dates contrôlées de début de processus de P_1	8.8	21.2	33.2	45.2	56.9	68.9	80.9	92.9	104.9	116.9
$y_1 = y_{122}$: dates de fin de P_1	11.8	24	36	48	60.3	72	84	96	108	120
u_2 : dates contrôlées de début de processus de P_2	11.8	24.8	37.8	50.8	63.9	76.9	89.9	102.9	115.9	128.9
$y_2 = y_{212}$: dates de fin de P_2	13.2	26	39	52	64.9	78	91	104	117	130

En dépit de l'erreur de modélisation sur les temps opératoires p_{122} et p_{212}, nous avons constaté (tableau 2.2) que la structure de commande par modèle interne permet d'avoir une sortie du système $Y = [y_1; y_2]$ égale ou proche de la référence $E = [e_1; e_2]$. Pendant la première et la cinquième période, la machine 2 n'est pas capable de fabriquer assez de produits afin de satisfaire le taux de production souhaité par la référence. Un tel problème cause une divergence entre la sortie du système et la référence (tableau 2.2).
Toutes ces méthodes de pilotage (commande en boucle ouverte, commande prédictive et commande par modèle interne) ne sont pas adaptées dans notre problématique de recherche pour des systèmes non linéaires.

En effet, ces techniques servent à résoudre des problèmes de suivi de trajectoire et de synchronisation afin de satisfaire les dates de besoin de produits sans tenir compte des conflits qui peuvent apparaître entre les opérations concurrentes sur les machines.

2.4.2 Pilotage des HVLV non libres avec ou sans maintenance

Après avoir discuter du problème de pilotage des systèmes HVLV à travers des structures de commande telles que la boucle ouverte et la CMI, nous nous intéressons maintenant à la commande des systèmes non libres. En effet, nous avons mis en évidence que la commande des systèmes libres se ramène à un problème conventionnel de suivi de trajectoire où les problèmes de conflit de concurrence entre opérations ne sont pas considérés. De part la spécificité des systèmes non libres où la présence des conflits entre opérations reste une caractéristique inhérente à leur fonctionnement, les techniques citées précédemment dans le cas libre s'avèrent non adaptées.

Dans le contexte non libre, le problème de commande est vu non pas comme une stratégie de suivi de trajectoire (uniquement) mais comme une méthodologie d'ordonnancement mixant une politique de suivi de trajectoire avec une stratégie d'optimisation non linéaire sous contraintes. Les modèles (2.6) et (2.11) proposés dans la section 2.3.2 contiennent deux variables d'entrée (de commande) : $U(T)$ et $V(T)$. La première commande sert à calculer la date d'entrée au plus tard de la matière première dans le système afin de satisfaire une production en juste-à-temps. Alors que la deuxième commande sert à déterminer l'ordre des opérations de produits et de maintenance sur les machines. Le problème de pilotage est abordé ici sous l'angle de l'ordonnancement prévisionnel. En effet un problème d'optimisation sous contraintes sera résolu afin de satisfaire les deux objectifs suivants :

1. Génération d'un ordonnancement faisable. Cette faisabilité sera déterminée par un choix adéquat des valeurs des variables de décision. Ceci sera expliqué dans le chapitre 3 (p.63 et 73).

2. La mesure des performances de production : le makespan et la somme des avances seront minimisés.

Génération d'un ordonnancement faisable

Dans les modèles (max,+) proposés (sans et avec maintenance) représentés par les équations (2.6) et (2.11), les variables de décision (de contrôle) $V(T)$ permettent à l'optimiseur d'allouer du temps dans l'horizon de planification de la production, en calculant les dates de débuts des opérations, pour fabriquer les produits (jobs J_i) sur les machines correspondantes. Elles sont intégrées dans le modèle afin de résoudre les conflits entre les opérations concurrentes sur les mêmes machines (opérations de produits O_{ijk} et tâches de maintenance PM_{hk}). Leurs valeurs déterminent les dates de début : $x_{ijk}(T)$ des opérations O_{ijk} et $x_{hk}(T)$ des activités de maintenance.
D'une part, afin d'obtenir des ordonnancements faisables et réalisables et pour éviter le chevauchement des opérations sur les machines, il faut avoir des bornes sur les variables de décision. Dans ce cas, des contraintes sur les variables de contrôle sont nécessaires pour les borner. Ces contraintes doivent satisfaire certaines conditions qui seront détaillées dans le chapitre 3 de cette thèse.
D'autre part, afin de respecter les contraintes de périodicité entre les tâches de maintenance dans les deux cas (maintenance répétitive périodique et maintenance flexible périodique), des relations entre les dates de début x_{hk} de PM_{hk} et les périodes $T_{(.)}$ seront établies dans l'algèbre (max,+). Ces contraintes seront aussi expliquées dans le troisième chapitre.

Mesure des performances de production : optimisation non linéaire sous contraintes

Comme indiqué dans la section 2.3.2, la représentation des systèmes HVLV (avec et sans maintenance) en utilisant l'algèbre (max,+) mène à un modèle non linéaire. Cette non linéarité est due à la multiplication entre les variables d'état $X(T)$ et les variables de contrôle $V(T)$. Elle sert à représenter le problème d'ordonnancement entre les opérations élémentaires et les activités de maintenance nécessitant un traitement sur les mêmes machines. D'après nos connaissances, il n'existe pas encore dans la littérature des techniques pour résoudre un modèle (max,+) non linéaire dans l'algèbre des dioïdes.

Toutes les techniques qui ont été développées telle que la *résiduation* sont utiles pour résoudre des équations linéaires dans le sens de l'algèbre des dioïdes. Pour toutes ces raisons le problème de pilotage (ordonnancement) des HVLV non libres avec ou sans maintenance est transformé en un problème d'optimisation non linéaire sous contraintes afin de minimiser séparément deux critères réguliers, à savoir :

- le makespan C_{max} : la minimisation de ce critère permet de finir l'ensemble des jobs J_i donné dans une période de temps courte et par la suite commencer l'exécution de la nouvelle liste de commandes le plus tôt possible. Par conséquent, nous améliorons l'occupation des machines. Dans ce cadre, le problème d'optimisation représenté sur la figure 2.8 peut être écrit comme suit :

$$\begin{cases} J^*(T) = \min\limits_{v_{(*)}(T)} J(T) = \min\limits_{v_{(*)}(T)} (C_{max}(T)) = \min\limits_{v_{(*)}(T)} (max(y_{ink}(T))) \\ sous\ contraintes\ non\ linéaires \end{cases} \tag{2.21}$$

avec $1 \leqslant i \leqslant l$, $1 \leqslant k \leqslant m$, $v_{(*)}(T)$ représente les variables de décision de la matrice de contrôle $V(T)$ qui sont les variables d'optimisation à déterminer et $y_{ink}(T) = x_{ink}(T) + p_{ink}(T)$ est la date de fin du job J_i pendant le cycle T.

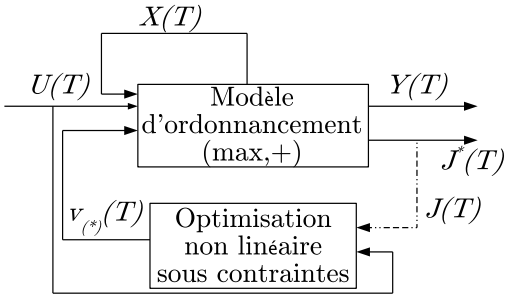

Figure 2.8 – Schéma de l'optimisation du makespan avec ou sans maintenance

- la somme des avances en juste à temps R : la minimisation de ce critère revient à résoudre un problème de poursuite de trajectoire en automatique conventionnelle (date de besoin de produits) en juste à temps. Soit $D_i(T)$ la date de besoin d'un job $J_i(i = 1 \ldots l)$ pour la $T^{ème}$ période. $R(T) = -\sum_{i=1}^{n} u_i(T)$: ce critère reflète l'effort du contrôle $u_i(T)$ (critère juste à temps). La minimisation de R mène à la maximisation des dates d'entrée $u_i(T)$ (variables d'optimisation à déterminer) de la matière première des jobs J_i dans le système. Par conséquent, la date de début de la première opération $x_{i1k}(T)$ de chaque job J_i sera égale à $u_i(T)$ ce qui permet de réduire les encours dans le système. Dans ce contexte, le problème d'optimisation donné par la figure 2.9 peut être écrit comme suit :

$$\begin{cases} J^*(T) = \min\limits_{v_{(*)}(T)} J(T) = \min\limits_{v_{(*)}(T)} R(T) \\ \textit{sous contraintes non linéaires} \end{cases} \tag{2.22}$$

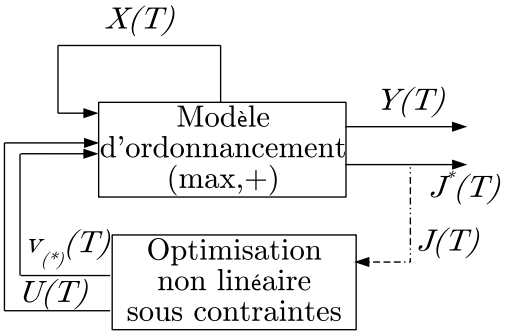

Figure 2.9 – Schéma de l'optimisation de l'avance totale avec ou sans maintenance

Le problème d'ordonnancement ainsi que les contraintes du problème d'optimisation seront détaillés et traités dans le chapitre 3.

2.5 Conclusion

Dans ce chapitre nous avons introduit l'algèbre (max, +) comme un outil permettant la représentation du comportement des systèmes HVLV. Nous avons pu montrer que ce type de systèmes peut être décrit par une représentation d'état. Cette dernière est linéaire dans le sens de (max, +) dans le cas des systèmes libres et non linéaire dans le cas des HVLV non libres.
Nous avons présenté des travaux portant sur la commande des SEDs dans l'algèbre (max, +). Les deux structures considérées sont la commande optimale en boucle ouverte, puis la commande en boucle fermée où l'objectif est de poursuivre une trajectoire en juste-à-temps. Nous avons également étendu ces différentes structures de commande aux systèmes HVLV à décision libre. Nous avons montré que le pilotage des systèmes HVLV non libres peut être vu comme un problème d'ordonnancement avec une optimisation non linéaire sous contraintes.
Dans le prochain chapitre, nous abordons le problème d'ordonnancement des systèmes HVLV.

Chapitre 3

Ordonnancement des systèmes HVLV à base de (max,+)

3.1 Introduction

Dans le chapitre précédent, nous avons montré que le comportement des systèmes HVLV non libres peut être décrit par une représentation d'état dans la structure (max,+). Cette dernière est donnée par les équations (2.6) et (2.11) dans les cas sans et avec maintenance. Des variables de décision ont été intégrées dans les deux cas afin de résoudre un problème d'ordonnancement sur les machines.
Ce chapitre a pour objectif de définir une méthodologie d'ordonnancement des systèmes HVLV sans et avec maintenance. Cette dernière est basée sur une stratégie de prise de décision. Pour ce faire, des contraintes sur les variables de décision sont d'abord établies afin d'avoir des ordonnancements faisables pour éviter le chevauchement entre les opérations de produits et les tâches de maintenance. Ensuite, un problème d'optimisation non-linéaire sous contraintes est résolu afin de minimiser certains critères de performance, à savoir : le makespan et l'avance totale en juste-à-temps. Des exemples illustratifs sont donnés afin de montrer la viabilité de la méthodologie proposée.

3.2 Notions générales d'ordonnancement

Définition 3.1. [88] L'ordonnancement consiste à organiser dans le temps la réalisation d'un ensemble de tâches, compte tenu de contraintes temporelles (délais, contraintes d'enchaînement, etc.) et de contraintes portant sur l'utilisation et la disponibilité des ressources requises par les tâches [89, 90]. Un ordonnancement décrit l'ordre d'exécution des tâches et l'allocation des ressources au cours du temps, afin de satisfaire un ou plusieurs critères d'optimisation.
Les ordres de fabrication (ou les *jobs*), suggérés par le calcul des besoins, représentent chacun une requête pour fabriquer une quantité déterminée de pièces pour une date donnée. Ils constituent les données d'entrée de l'ordonnancement et permettent de définir, au

moyen des gammes de fabrication, l'ensemble des tâches que la fonction ordonnancement doit planifier. Une tâche est localisée dans le temps par une date de début et une durée ou une date de fin. Elle utilise une ou plusieurs ressources. En sortie de la fonction d'ordonnancement, on obtient un planning qui restitue l'affectation des tâches fournies en entrée à des dates précises pour des durées déterminées sur les différentes ressources. Ce planning cherche à satisfaire des objectifs, en respectant le plus possible les contraintes précitées [91].
Les problèmes d'ordonnancement ont de nombreux champs d'application. On peut distinguer par exemple les applications liées à la production de biens, celles dans les systèmes informatiques (tâches : jobs ; ressources : processeurs ou mémoire ...) et celles dans l'industrie (gestion de projet, problèmes d'ateliers) [92]. Le problème d'ordonnancement des ateliers par exemple, constitue sûrement une des difficultés importantes des systèmes de gestion et de conduite pour les entreprises. En effet, c'est à ce niveau que doivent être prises en compte les caractéristiques réelles multiples et complexes des ateliers, ainsi que les perturbations qui viennent les modifier, aussi bien internes (pannes des machines, etc.) qu'externes (perturbation de la demande, etc.).
Dans un atelier, une production consiste à fabriquer un produit ou un lot de produits ; l'ordre de passage de ces produits sur un ensemble de machines ainsi que les temps élémentaires de processus est appelé « gamme ». La réalisation d'un job nécessite plusieurs opérations élémentaires appelées « tâches ». Les opérations seront effectuées sur les machines. Le rôle essentiel de l'ordonnancement est de programmer l'exécution d'une réalisation en attribuant des ressources aux tâches et de fixer leurs dates d'exécution. L'ordonnancement d'atelier consiste à affecter plusieurs tâches à des moyens de fabrication afin de réaliser des travaux en respectant les contraintes de fabrication et en optimisant un ou plusieurs critères.
Les éléments principaux constituant un problème d'ordonnancement sont donc les tâches, les ressources, les contraintes, et la fonction critère. Avant d'aborder la résolution du problème d'ordonnancement, nous allons définir l'ensemble des éléments nécessaires pour la compréhension du problème traité.

3.3 Éléments d'un problème d'ordonnancement

D'après [14, 91], un ordonnancement est constitué de quatre éléments :
- les opérations,
- les ressources,
- les contraintes,
- les critères.

3.3.1 Les opérations

Ce sont les opérations élémentaires d'un produit à effectuer lors de sa fabrication. Une opération j d'un produit i sur une machine k appelée O_{ijk} est généralement caractérisée

par une date de début x_{ijk} et une date de fin c_{ijk}. On peut également associer une date de besoin D_i à un produit i au delà de laquelle il n'est plus possible de l'exécuter. Si l'ensemble des opérations à exécuter au cours du temps est donné a priori, le problème d'ordonnancement est dit « statique ». Si l'ensemble des opérations, ou des tâches à exécuter, évolue avec le temps, éventuellement de façon non déterministe, alors le problème est « dynamique ». Dans ce cas, la résolution peut être envisagée à travers une suite de résolution de problèmes statiques, chaque phase débutant par une prise d'informations permettant d'actualiser le modèle du problème à résoudre [93]. Dans cette thèse, nous nous intéressons à l'ordonnancement « statique » des systèmes HVLV.
Une opération peut se réaliser par morceaux, il s'agit d'une opération préemptive, ou sans interruption, on parle alors d'une tâche non-préemptive. Certaines opérations sont préemptives répétées, ç-à-d, l'interruption de ces tâches est autorisée, mais l'exécution doit être recommencée depuis le début. Nous ne considérons dans cette thèse que les tâches non-préemptives.

3.3.2 Les ressources

L'exécution des différentes tâches nécessite la mise en œuvre d'un ensemble de moyens techniques et d'opérateurs humains. La capacité d'une ressource est en réalité limitée. Nous tenons compte de cette limitation dans ce chapitre en supposant que les ressources sont sujettes à des pannes et nécessitent des interventions de maintenance.
On distingue deux types de ressources, les ressources renouvelables et celles non renouvelables.

- Les ressources renouvelables : une ressource est dite renouvelable si, après avoir été allouée à une tâche, elle redevient disponible à la fin de l'exécution de cette dernière. Les ressources renouvelables usuelles sont les machines, les processeurs, les fichiers, le personnel,... etc.
- Les ressources non renouvelables (consommables) : ces ressources ne sont plus disponibles après la fin de la tâche. Elles sont en fait épuisées par la réalisation de cette opération. C'est le cas pour l'argent, la matière première,... etc.

3.3.3 Les contraintes

Les contraintes expriment des restrictions sur les valeurs que peuvent avoir les variables de décision. Leur prise en compte permet d'avoir un ordonnancement faisable et réalisable. Les problèmes d'ordonnancement sont rendus difficiles à cause des contraintes à respecter. Parmi ces contraintes on distingue, les contraintes potentielles ou contraintes de précédence, les contraintes de concurrence, les contraintes intérieures et les contraintes extérieures :

1. Contraintes potentielles : elles lient les tâches entre elles, c'est-à-dire que telle opération ne peut être exécutée avant telle autre. Elles peuvent être temporelles (quand il faut attendre la fin d'une tâche avant de lancer la suivante). Ce sont des contraintes

qui vont définir les gammes de production.

2. Contraintes de concurrence : ce type de contraintes est lié aux ressources utilisées par les tâches. Ces contraintes apparaîtront lorsque deux tâches, utilisant la même machine, ne pourront pas s'exécuter simultanément.
3. Contraintes intérieures : ce sont des conditions directement liées au système de production et à ses performances. Nous pouvons citer :
   (a) les capacités des machines et des moyens de transport ;
   (b) les dates de disponibilités de la matière première des produits et des moyens de transport ;
4. Contraintes extérieures : ce sont des conditions imposées extérieurement. Elles sont indépendantes du système de production, telles que :
   (a) les dates de besoin de produits, imposées généralement par les commandes du client ;
   (b) les niveaux de priorité et d'urgence de quelques commandes et de quelques clients ;
   (c) les retards accordés pour certains produits.

3.3.4 Les critères d'ordonnancement

Lorsqu'on aborde un problème d'ordonnancement, il est crucial de définir un certain nombre d'objectifs (critères) à atteindre. Il s'agit ici d'optimiser (maximiser ou minimiser) une fonction d'évaluation en respectant un certain nombre de contraintes. Les trois piliers d'un ordonnancement sont : le coût, la qualité et le délai. Toute évaluation d'un ordonnancement n'aura donc de sens aux yeux de la production que dans la mesure où elle porte sur ces trois facettes.
On distingue deux types de critères à optimiser : les critères liés au temps et ceux liés aux coûts [94]. Dans le premier cas, on trouve, par exemple le critère qui mesure la date d'achèvement de l'ensemble de l'ordonnancement ou le critère qui mesure l'avance des travaux par rapport à leur date de fin souhaitée. Dans le second cas, on peut citer les critères qui représentent un coût d'utilisation des machines et ceux qui représentent un coût lié à l'attente des opérations avant et/ou après leur traitement. Les critères les plus utilisés pour évaluer la qualité d'un ordonnancement sont la durée totale d'exécution (makespan), les dates de fin au plus tard et le coût d'exécution :

- La durée totale : la durée totale de l'ordonnancement notée C_{max} est égale à la date d'achèvement de la tâche la plus tardive.
- Les dates de fin au plus tard : dans beaucoup de problèmes réels, il faut respecter au mieux les délais, en l'occurrence les dates de fin au plus tard. Ceci se fait en minimisant soit la plus grande avance R_{max}, soit la somme des avances R.
- La minimisation d'un coût : est un critère qui peut s'exprimer sous des formes très variées comme par exemple la minimisation des encours, la minimisation du coût du stockage de la matière première,... etc.

Dans la littérature, le critère le plus utilisé le "makespan". C'est-à-dire sur la minimisation de la différence entre l'instant où le dernier produit à fabriquer sort et l'instant où le premier produit entre dans l'atelier. Dans ce chapitre, deux critères seront utilisés pour quantifier la performance de notre approche, à savoir : C_{max} et R en juste-à-temps.

3.4 État de l'art sur l'ordonnancement des systèmes HVLV

Un état de l'art détaillé sur les méthodes de résolution des systèmes HVLV de type job shop a été publié par Jain et Meeran en 1998 [95]. Ces auteurs ont étudié les différents algorithmes exacts et les méthodes approchées de résolution publiés depuis les années 50. Nous allons classer cet état de l'art en deux catégories : les méthodes exactes et les méthodes approchées.

3.4.1 Les méthodes exactes

Akers et Freidman [96] sont probablement les premiers à avoir étudié le problème d'ordonnancement des systèmes job shop. Ces auteurs ont proposé une méthode de résolution qui utilise l'algèbre de Boole. Jackson [97] a adapté ensuite l'algorithme de Jhonson [98] pour résoudre un problème d'ordonnancement d'un flow shop. Roy et Sussman [99] sont les premiers qui ont utilisé le graphe de précédence disjonctif.
Plusieurs méthodes efficaces permettant de résoudre un problème d'ordonnancement dans un job shop ont été proposés. Parmi ces méthodes, on trouve la stratégie de séparation et d'évaluation (Branch and Bound). Carlier et Pinson ont pu résoudre avec cette méthode exacte un problème d'ordonnancement de taille 10×10 [100]. Les mêmes auteurs ont réussi, une année plus tard, de résoudre des problèmes plus complexes [101].

3.4.2 Les méthodes approchées

Ces approches sont largement utilisés pour résoudre des problèmes de grande taille bien qu'elles ne permettent pas de garantir l'obtention de la solution optimale. Les premiers heuristiques sont les règles d'affectation [102] qui utilisent les caractéristiques du problème (les charges des machines, les temps opératoires des opération, etc.) pour affecter une priorité d'exécution sur les machines. Cette méthode semble facile à appliquer mais son champ d'application est limité.
Adams et *al* ont proposé un algorithme approximatif [103] pour résoudre un problème de job shop en traitant les machines consécutivement dans un ordre de priorité décroissant. Par conséquent, la machine ayant l'ordre le plus grand est sélectionnée. A chaque fois, une seule machine est considérée en affectant d'une manière optimale ses opérations sans tenir compte des autres machines.
Les débuts des années 90 ont connu l'utilisation de la recherche tabou [104] et la simulation [105], pour la résolution d'un job shop. Récemment, les chercheurs se sont intéressés

aux réseaux de neurones [106], aux colonies de fourmi [107] et aux algorithmes génétiques [10–12].

3.4.3 Discussion

Malgré leur capacité d'apprentissage et de généralisation, les réseaux de neurones ne fournissent pas des résultats performants pour la résolution d'un job shop. Ceci est dû à la difficulté de trouver une bonne fonction d'activation des neurones [106].
Des travaux utilisant les colonies de fourmi [107] pour résoudre un job shop présentent des résultats avec des erreurs variant entre 8 et 32 par rapport à l'optimum.
Malgré leur adaptabilité à la résolution des problèmes de job shop, les algorithmes génétiques classiques et modifiés [11, 12] ne sont pas facile à appliquer. Leur application demande une grande expertise afin de régler plusieurs paramètres (tels que le taux de croisement et le taux de mutation) afin de converger vers la solution optimale.
Les méthodes exactes qui se basent sur des formalismes mathématiques tels que l'algèbre (max, +) restent une bonne solution qui est facile à appliquer surtout dans le cas des systèmes HVLV dû à la faible cadence de production. Ces méthodes ne nécessitent pas une grande expertise comme les méthodes approchées et garantissent des solutions optimales. Toutefois, les méthodes exactes restent redevables et dépendantes d'un certain nombre de conditions (suppositions) nécessaires pour l'obtention du modèle et à la résolution du problème d'ordonnancement associé. Malgré ces insuffisances, dans ce travail nous adoptons une méthode exacte basé sur l'optimisation non linéaire afin de résoudre un problème d'ordonnancement des systèmes HVLV.

3.5 Ordonnancement des systèmes HVLV sans maintenance

3.5.1 Génération d'un ordonnancement faisable

Après avoir proposé et développé, dans le chapitre 2, le modèle (max,+) non-linéaire et non libre d'un système HVLV où l'introduction de nouvelles variables (entrées de contrôle) de décision pour pouvoir résoudre le problème d'ordonnancement est la clef de notre approche, nous discutons dans cette section la spécification des contraintes à respecter sur ces variables de décision $v_{ijk;i''j''k}(T+1)$ $(i \neq i'', j \neq j'')$ pour résoudre notre problématique d'ordonnancement. Afin d'illustrer la démarche de synthèse proposée, deux exemples d'illustration, simples et académiques seront utilisés pour guider le lecteur.

Développement des contraintes sur les variables de décision

Comme indiqué dans le chapitre 2, les variables de décision $v_{ijk;i''j''k}(T+1)$ sont introduites dans les équations (2.7) et (2.9) (p.34-35) afin de résoudre les conflits entre les opérations concurrentes sur les mêmes machines pendant la période $T+1$. Leurs valeurs déterminent les dates de début $x_{ijk}(T+1)$ des opérations O_{ijk}.

D'une part, si $v_{ijk;i''j''k}(T+1) = -\infty$, alors l'opération O_{ijk} est exécutée avant $O_{i''j''k}$ sur la machine M_k pendant le cycle $T+1$. D'autre part, si $v_{ijk;i''j''k}(T+1) = 0$, alors O_{ijk} est traitée après $O_{i''j''k}$ sur M_k pendant la période $T+1$.
Afin d'obtenir des ordonnancements faisables et réalisables et pour éviter le chevauchement des opérations sur les machines, des bornes sur les variables de décision sont nécessaires. Ces contraintes doivent satisfaire les conditions suivantes pour chaque paire d'opérations concurrentes O_{ijk} et $O_{i''j''k}$ $(i \neq i'', j \neq j'')$:

1. Soit $v_{ijk;i''j''k}(T+1)$ ou $v_{i''j''k;ijk}(T+1)$ est égale à $-\infty$.
2. Si $v_{ijk;i''j''k}(T+1)$ est égale à $-\infty$, alors $v_{i''j''k;ijk}(T+1)$ est égale à zéro.
3. Si $v_{i''j''k;ijk}(T+1)$ est égale à $-\infty$, alors $v_{ijk;i''j''k}(T+1)$ est égale à zéro.

Ces contraintes peuvent s'exprimer comme suit :

$$\begin{cases} v_{ijk;i''j''k}(T+1) + v_{i''j''k;ijk}(T+1) = -\infty \\ max(v_{ijk;i''j''k}(T+1); v_{i''j''k;ijk}(T+1)) = 0 \end{cases} \tag{3.1}$$

En utilisant les notations de l'algèbre (max,+), l'équation (3.1) devient :

$$\begin{cases} v_{ijk;i''j''k}(T+1) \otimes v_{i''j''k;ijk}(T+1) = -\infty \\ v_{ijk;i''j''k}(T+1) \oplus v_{i''j''k;ijk}(T+1) = 0 \end{cases} \tag{3.2}$$

Exemple illustratif 1

Nous considérons dans cet exemple deux types de produits (jobs) J_1 et J_2 qui sont fabriqués périodiquement sur une seule machine c-à-d le même motif (ordre des opérations) est répété sur chaque période T. Soient p_{111} et p_{211} les temps opératoires de J_1 et J_2. Les variables d'état de ce système pendant la période $T+1$ sont comme suit :

- $x_{111}(T+1)$ est la date de début du job J_1 sur la machine M_1.
- $x_{211}(T+1)$ est la date de début du job J_2 sur la machine M_1.

La dynamique du système ainsi que les contraintes de ce problème sont décrites par les équations (max,+) suivante :

$$x_{111}(T+1) = max(y_{111}(T), y_{211}(T), u_1(T+1), x_{211}(T+1) + p_{211} + v_{111;211}(T+1))$$

$$x_{211}(T+1) = max(y_{111}(T), y_{211}(T), u_2(T+1), x_{111}(T+1) + p_{111} + v_{211;111}(T+1))$$

$$y_{111}(T+1) = x_{111}(T+1) + p_{111}$$

$$y_{211}(T+1) = x_{211}(T+1) + p_{211}$$

$$v_{111;211}(T+1) + v_{211;111}(T+1) = -\infty$$

$$max(v_{111;211}(T+1); v_{211;111}(T+1)) = 0$$

Et en utilisant les notations de l'algèbre (max,+) :

$$x_{111}(T+1) = y_{111}(T) \oplus y_{211}(T) \oplus u_1(T+1) \oplus x_{211}(T+1) \otimes p_{211} \otimes v_{111;211}(T+1)$$

$$x_{211}(T+1) = y_{111}(T) \oplus y_{211}(T) \oplus u_2(T+1) \oplus x_{111}(T+1) \otimes p_{111} \otimes v_{211;111}(T+1)$$

$$y_{111}(T+1) = x_{111}(T+1) \otimes p_{111}$$

$$y_{211}(T+1) = x_{211}(T+1) \otimes p_{211}$$

$$v_{111;211}(T+1) \otimes v_{211;111}(T+1) = \epsilon$$

$$v_{111;211}(T+1) \oplus v_{211;111}(T+1) = e$$

En choisissant des valeurs différentes pour les variables de décision qui vérifient les contraintes représentées par l'équation (3.2), l'ordonnancement sur la machine M_1 peut être déterminé. Pour illustrer, nous supposons que $y_{111}(T) \oplus y_{211}(T) = 0, u_1(T+1) = u_2(T+1) = 3, p_{111} = 2, p_{211} = 4$.

Cas 1 : $(v_{111;211}(T+1), v_{211;111}(T+1)) = (\epsilon, e)$
Ces valeurs choisies pour les variables de décision donnent les dates de début et de fin des opérations O_{111} et O_{211} pendant la période T suivantes :

$$\begin{aligned} x_{111}(T+1) &= y_{111}(T) \oplus y_{211}(T) \oplus u_1(T+1) \oplus x_{211}(T+1) \otimes p_{211} \otimes v_{111;211}(T+1) \\ &= 0 \oplus 0 \oplus 3 \oplus x_{211}(T+1) \otimes 4 \otimes \epsilon \\ &= 3 \oplus \epsilon = 3 \end{aligned}$$

$$\begin{aligned} x_{211}(T+1) &= y_{111}(T) \oplus y_{211}(T) \oplus u_2(T+1) \oplus x_{111}(T+1) \otimes p_{111} \otimes v_{211;111}(T+1) \\ &= 0 \oplus 0 \oplus 3 \oplus 3 \otimes 2 \otimes 0 \\ &= 0 \oplus 3 \oplus 5 = 5 \end{aligned}$$

$$y_{111}(T+1) = x_{111}(T+1) \otimes p_{111} = 3 \otimes 2 = 5$$

$$y_{211}(T+1) = x_{211}(T+1) \otimes p_{211} = 5 \otimes 4 = 9$$

Cette situation correspond au Gantt représenté par la figure 3.1 :

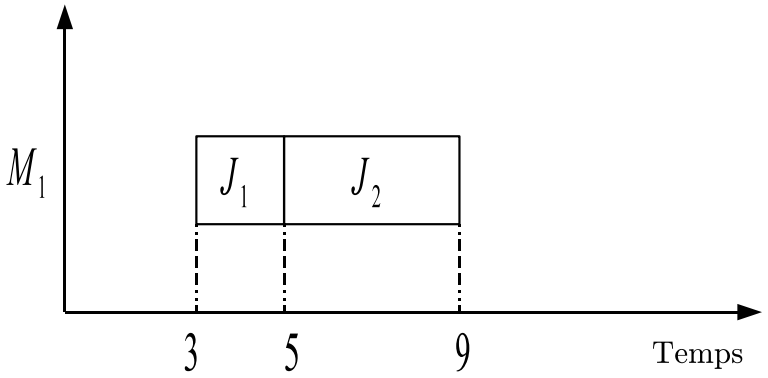

Figure 3.1 – Diagramme de Gantt du cas 1 de l'exemple 1

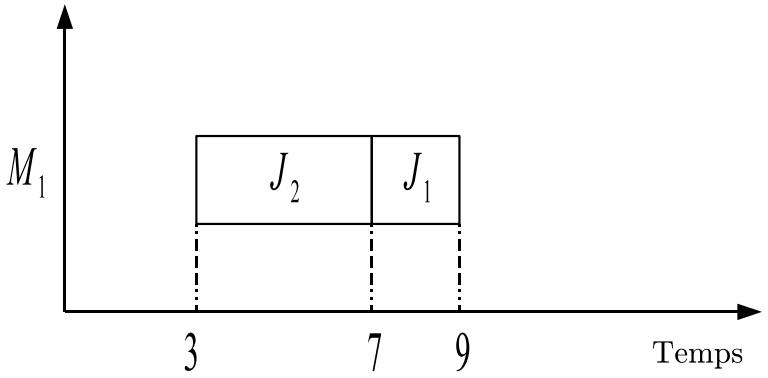

Figure 3.2 – Diagramme de Gantt du cas 2 de l'exemple 1

Cas 2 : $(v_{111;211}(T+1), v_{211;111}(T+1)) = (e, \epsilon)$
Ces valeurs choisies pour les variables de décision donnent les dates de début et de fin des opérations O_{111} et O_{211} suivantes :

$$\begin{aligned} x_{211}(T+1) &= y_{111}(T) \oplus y_{211}(T) \oplus u_2(T+1) \oplus x_{111}(T+1) \otimes p_{111} \otimes v_{211;111}(T+1) \\ &= 0 \oplus 0 \oplus 3 \oplus x_{111}(T+1) \otimes 2 \otimes \epsilon \\ &= 0 \oplus 3 \oplus \epsilon = 3 \end{aligned}$$

$$\begin{aligned} x_{111}(T+1) &= y_{111}(T) \oplus y_{211}(T) \oplus u_1(T+1) \oplus x_{211}(T+1) \otimes p_{211} \otimes v_{111;211}(T+1) \\ &= 0 \oplus 0 \oplus 3 \oplus x_{211}(T+1) \otimes 4 \otimes 0 \\ &= 3 \oplus 3 \otimes 4 = 3 \oplus 7 = 7 \end{aligned}$$

$$y_{111}(T+1) = x_{111}(T+1) \otimes p_{111} = 7 \otimes 2 = 9$$

$$y_{211}(T+1) = x_{211}(T+1) \otimes p_{211} = 3 \otimes 4 = 7$$

Cette situation correspond au Gantt représenté par la figure 3.2 :

Exemple illustratif 2

Nous considérons dans cet exemple un système HVLV avec deux machines M_1 et M_2 et deux types de produits à fabriquer J_1 et J_2. Les temps opératoires sont donnés dans le tableau 3.1 :

Tableau 3.1 – Données de production de l'exemple 2

	M_1	M_2
J_1	$p_{111} = 2$	$p_{122} = 5$
J_2	$p_{211} = 3$	$p_{222} = 8$

Les variables d'état de ce système pendant la période $(T+1)$ sont comme suit :

- $x_{111}(T+1)$ est la date de début de l'opération O_{111} sur la machine M_1.
- $x_{122}(T+1)$ est la date de début de l'opération O_{122} sur la machine M_2.
- $x_{211}(T+1)$ est la date de début de l'opération O_{211} sur la machine M_1.
- $x_{222}(T+1)$ est la date de début de l'opération O_{222} sur la machine M_2.

La dynamique du système ainsi que les contraintes de ce problème sont décrites par les équations (max,+) suivantes :

$$x_{111}(T+1) = max(y_{111}(T), y_{211}(T), u_1(T+1), x_{211}(T+1) + p_{211} + v_{111;211}(T+1))$$

$$x_{122}(T+1) = max(y_{122}(T), y_{222}(T), x_{111}(T+1) + p_{111}, x_{222}(T+1) + p_{222} + v_{122;222}(T+1))$$

$$x_{211}(T+1) = max(y_{111}(T), y_{211}(T), u_2(T+1), x_{111}(T+1) + p_{111} + v_{211;111}(T+1))$$

$$x_{222}(T+1) = max(y_{122}(T), y_{222}(T), x_{211}(T+1) + p_{211}, x_{122}(T+1) + p_{122} + v_{222;122}(T+1))$$

$$y_{111}(T+1) = x_{111}(T+1) + p_{111}$$

$$y_{122}(T+1) = x_{122}(T+1) + p_{122}$$

$$y_{211}(T+1) = x_{211}(T+1) + p_{211}$$

$$y_{222}(T+1) = x_{222}(T+1) + p_{222}$$

$$v_{111;211}(T+1) + v_{211;111}(T+2) = -\infty$$

$$max(v_{111;211}(T+1); v_{211;111}(T+1)) = 0$$

$$v_{122;222}(T+1) + v_{222;122}(T+1) = -\infty$$

$$max(v_{122;222}(T+1); v_{222;122}(T+1)) = 0$$

Et en utilisant les notations de l'algèbre (max,+) :

$$x_{111}(T+1) = y_{111}(T) \oplus y_{211}(T) \oplus u_1(T+1) \oplus x_{211}(T+1) \otimes p_{211} \otimes v_{111;211}(T+1)$$

$$x_{122}(T+1) = y_{122}(T) \oplus y_{222}(T) \oplus x_{111}(T+1) \otimes p_{111} \oplus x_{222}(T+1) \otimes p_{222} \otimes v_{122;222}(T+1)$$

$$x_{211}(T+1) = y_{111}(T) \oplus y_{211}(T) \oplus u_2(T+1) \oplus x_{111}(T+1) \otimes p_{111} \otimes v_{211;111}(T+1)$$

$$x_{222}(T+1) = y_{122}(T) \oplus y_{222}(T) \oplus x_{211}(T+1) \otimes p_{211} \oplus x_{122}(T+1) \otimes p_{122} \otimes v_{222;122}(T+1)$$

$$y_{111}(T+1) = x_{111}(T+1) \otimes p_{111}$$

$$y_{122}(T+1) = x_{122}(T+1) \otimes p_{122}$$

$$y_{211}(T+1) = x_{211}(T+1) \otimes p_{211}$$

$$y_{222}(T+1) = x_{222}(T+1) \otimes p_{222}$$

$$v_{111;211}(T+1) \otimes v_{211;111}(T+2) = \epsilon$$

$$v_{111;211}(T+1) \oplus v_{211;111}(T+1) = e$$

$$v_{122;222}(T+1) \otimes v_{222;122}(T+1) = \epsilon$$

$$v_{122;222}(T+1) \oplus v_{222;122}(T+1)) = e$$

En choisissant des valeurs différentes pour les variables de décision qui vérifient les contraintes représentées par l'équation (3.2), l'ordonnancement sur les machines M_1 et M_2 peut être déterminé. Pour illustrer, nous supposons que $y_{111}(T) \oplus y_{211}(T) = 5, y_{122}(T) \oplus y_{222}(T) = 7, u_1(T+1) = 2$ et $u_2(T+1) = 7$.

Cas 1 : $(v_{111;211}(T+1), v_{211;111}(T+1), v_{122;222}(T+1), v_{222;122}(T+1)) = (\epsilon, e, \epsilon, e)$
En effectuant le même calcul (max,+) de l'exemple 1, les valeurs choisies des variables de décision donnent les dates de début et de fin des opérations $O_{111}, O_{122}, O_{211}$ et O_{222} suivantes :

$$x_{111}(T+1) = 5 \oplus 2 \oplus x_{211}(T+1) \otimes 3 \otimes \epsilon = 5$$

$$x_{122}(T+1) = 7 \oplus 5 \otimes 2 \oplus x_{222}(T+1) \otimes 8 \otimes \epsilon = 7$$

$$x_{211}(T+1) = 5 \oplus 7 \oplus 5 \otimes 2 \otimes 0 = 7$$

$$x_{222}(T+1) = 7 \oplus 7 \otimes 3 \oplus 7 \otimes 5 \otimes 0 = 12$$

$$y_{111}(T+1) = 5 \otimes 2 = 7$$

$$y_{211}(T+1) = 7 \otimes 3 = 10$$

$$y_{122}(T+1) = 7 \otimes 5 = 12$$

$$y_{222}(T+1) = 12 \otimes 8 = 20$$

Cette situation correspond au Gantt représenté par la figure 3.3 :

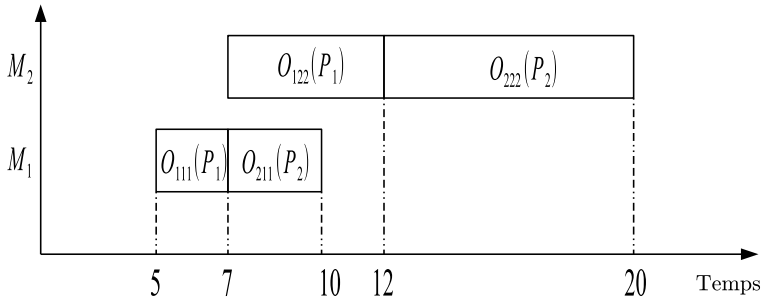

Figure 3.3 – Diagramme de Gantt du cas 1 de l'exemple 2

Cas 2 : $(v_{111;211}(T+1), v_{211;111}(T+1), v_{122;222}(T+1), v_{222;122}(T+1)) = (e, \epsilon, e, \epsilon)$
En effectuant le même calcul (max,+) de l'exemple 1, les valeurs choisies des variables de décision donnent les dates de début et de fin des opérations $O_{111}, O_{122}, O_{211}$ et O_{222} suivantes :

$$x_{111}(T+1) = 10, x_{122}(T+1) = 18, x_{211}(T+1) = 7, x_{222}(T+1) = 10, y_{111}(T+1) = 12$$

$$y_{211}(T+1) = 10, y_{122}(T+1) = 23, y_{222}(T+1) = 18$$

Cette situation correspond au Gantt représenté par la figure 3.4 :

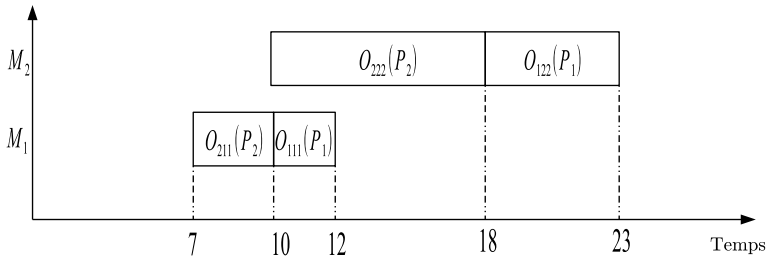

Figure 3.4 – Diagramme de Gantt du cas 2 de l'exemple 2

Les deux exemples présentés dans cette section montrent bien que le modèle (max,+) non linéaire proposé est capable de générer tous les ordonnancements faisables et réalisables. Nous pouvons remarquer d'après les figures 3.3 et 3.4 que la valeur du makespan $C_{max}(T+1)$ change selon les valeurs des variables de décision choisies (selon l'ordre des opérations sur les machines). En effet, dans le premier cas de l'exemple 2, le makespan est égal à 20 unités de temps pendant la période $T+1$, alors que sa valeur dans le deuxième cas est 23 unités de temps. Par conséquent, les valeurs des variables de décision sont choisies adéquatement afin d'atteindre un objectif souhaité et pour optimiser un critère de performance spécifique.

Cas 3 : $(v_{111;211}(T+1), v_{211;111}(T+1), v_{122;222}(T+1), v_{222;122}(T+1)) = (\epsilon, e, e, \epsilon)$

$$x_{111}(T+1) = 5, x_{122}(T+1) = 18, x_{211}(T+1) = 7, x_{222}(T+1) = 10, y_{111}(T+1) = 7$$

$$y_{211}(T+1) = 10, y_{122}(T+1) = 23, y_{222}(T+1) = 18$$

Cette situation correspond au Gantt représenté par la figure 3.5 :

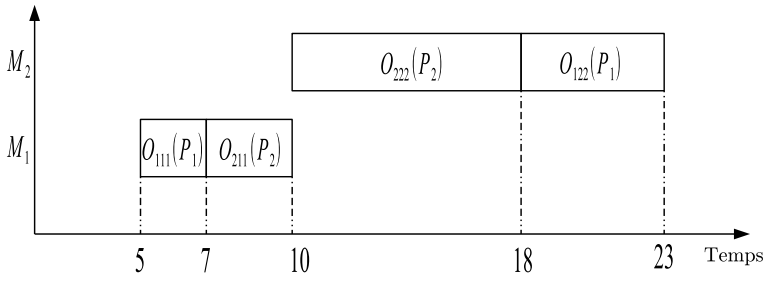

Figure 3.5 – Diagramme de Gantt du cas 3 de l'exemple 2

Cas 4 : $(v_{111;211}(T+1), v_{211;111}(T+1), v_{122;222}(T+1), v_{222;122}(T+1)) = (e, \epsilon, \epsilon, e)$

$$x_{111}(T+1) = 10, x_{122}(T+1) = 12, x_{211}(T+1) = 7, x_{222}(T+1) = 17, y_{111}(T+1) = 12$$

$$y_{211}(T+1) = 10, y_{122}(T+1) = 17, y_{222}(T+1) = 25$$

Cette situation correspond au Gantt représenté par la figure 3.6 :

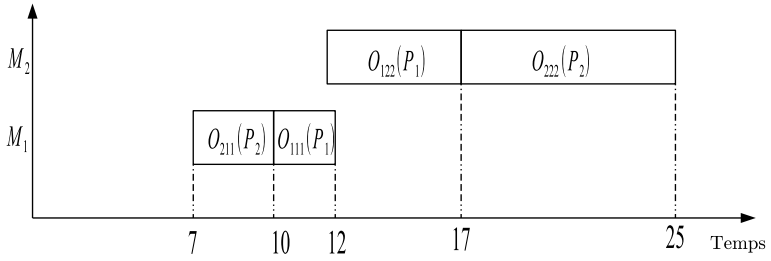

Figure 3.6 – Diagramme de Gantt du cas 4 de l'exemple 2

3.5.2 Mesure des performances de production : optimisation non linéaire sous contraintes

Nous avons proposé dans la section 2.4.2 du chapitre 2, la formalisation du problème d'optimisation pour l'ordonnancement des systèmes HVLV. Nous considérons dans ce paragraphe une optimisation sur un seul cycle de production, à savoir pendant la période $T+1$. Par abus de notation et pour alléger les écritures, la période T sera omise dans toutes les équations. La date de disponibilité des machines dans un nouveau cycle de production sera remplacée par la variable t au lieu de $y_{ijk}(T)$ qui sera la date de déclenchement de l'ordonnancement pendant un nouveau horizon de planification. Nous discuterons les

contraintes d'optimisation dans le cas d'un ordonnancement sans maintenance et nous donnerons par la suite quelques exemples illustratifs.

Minimisation du makespan

Le problème d'optimisation du makespan C_{max} représenté sur la figure 2.8 (cf. chapitre 2, p.54) est décrit par l'équation (2.21). Les variables d'optimisation sont les variables de décision $v_{ijk;i''j''k}\ \forall i, i'' = 1 \dots l,\ j, j'' = 1 \dots n,\ i \neq i''$ et $j \neq j''$. Nous supposons que la matière première de chaque produit J_i est disponible à l'instant 0, c-à-d, $u_i = 0\ \forall i = 1 \dots l$ et que $t = 0$. Dans ce contexte, les contraintes de ce problème d'optimisation sont décrites par les équations suivantes :

$$x_{ijk} = max(t; x_{i''j''k} + p_{i''j''k} + v_{ijk;i''j''k}; u_i) \tag{3.3}$$

$$x_{ijk} = max(t; x_{i(j-1)t} + p_{i(j-1)t}; x_{i''j''k} + p_{i''j''k} + v_{ijk;i''j''k}) \tag{3.4}$$

$$v_{ijk;i''j''k} + v_{i''j''k;ijk} = -\infty \tag{3.5}$$

$$max(v_{ijk;i''j''k}; v_{i''j''k;ijk}) = 0 \tag{3.6}$$

Les contraintes (3.3), (3.4) et (3.6) sont non linéaires et non convexes dû à la relation *max* entre les différentes variables.

Exemple illustratif : Dans toute la suite, l'exemple représenté sur le tableau 3.1 p.66 est exploité. L'algorithme d'optimisation sous contrainte décrit par l'équation (2.21) est appliqué sur une seule période afin de minimiser le makespan C_{max}. La méthode proposée est implémentée en utilisant un logiciel d'optimisation non linéaire appelé *LINGO*. La valeur optimale obtenue du makespan est $C^*_{max} = 15$ unités de temps (voir figure 3.7). Le diagramme de Gantt présenté à la figure 3.7 montre l'enchainement des opérations et les dates de fin C_i des différents jobs J_i $(i = 1 \dots 2)$ récapitulées dans le tableau 3.2.

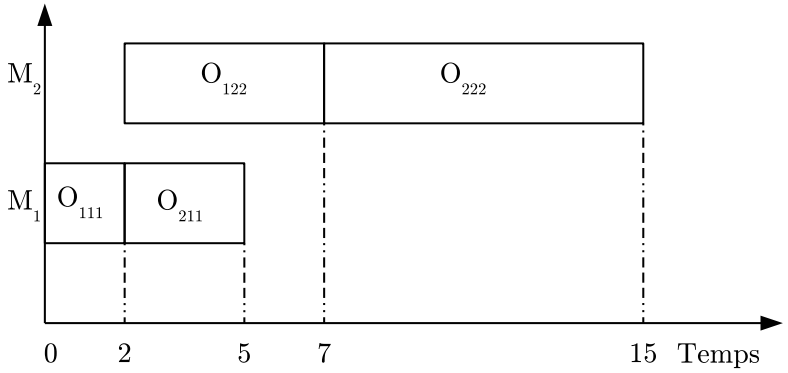

Figure 3.7 – Ordonnancement des opérations sur les machines

Tableau 3.2 – Dates de fin des produits

Jobs	J_1	J_2
C_i	7	**15**

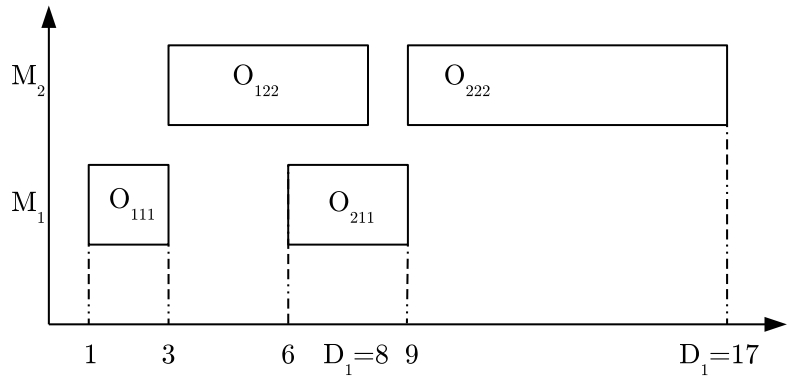

Figure 3.8 – Ordonnancement des opérations sur les machines

Minimisation de l'avance totale en juste-à-temps

Le problème d'optimisation de l'avance totale en juste-à-temps R représenté sur la figure 2.8 est décrit par l'équation (2.21) (cf. chapitre 2). Les variables d'optimisation sont les variables de décision $v_{ijk;i"j"k}$ $\forall i, i" = 1 \dots l, j, j" = 1 \dots n, i \neq i"$ et $j \neq j"$ et les dates d'entrée des produits dans le système u_i. Nous supposons que $t = 0$. Dans ce contexte, les contraintes de ce problème d'optimisation sont décrites par les équations (3.3), (3.4), (3.5), (3.6) et aussi par les équations suivantes :

$$x_{ijk} \geqslant u_i \tag{3.7}$$

$$y_{ijk} = x_{ijk} + p_{ijk} \leqslant D_i \tag{3.8}$$

avec D_i sont les dates de besoin des produits P_i pendant l'horizon de planification.

Exemple illustratif : Nous considérons ici l'exemple représenté par le tableau 3.1. L'algorithme d'optimisation sous contraintes décrit par l'équation (2.22) est appliqué sur une seule période afin de minimiser l'avance totale R. La méthode proposée est implémentée en utilisant un logiciel d'optimisation non linéaire appelé *LINGO*. Soit D_i les dates de besoin des jobs $J_i (i = 1 \dots 2)$ (Tableau 3.3).

Tableau 3.3 – Dates de besoin D_i des produits

Jobs	J_1	J_2
D_i	8	17

La valeur optimale obtenue de l'avance totale est $R = -7$ et (figure 3.8). Le diagramme de Gantt présenté à la figure 3.8 montre l'enchainement des opérations des différents jobs J_i $(i = 1 \dots 2)$.

Les tableaux 3.4 et 3.5 montrent que le critère de la production en juste à temps est satisfait. En effet, les dates de début de la première opération x_{i1k} de chaque job J_i est inférieure ou égale à u_i la date d'entrée de chaque produit dans le système. Par conséquent la minimisation du critère proposé R entraîne une maximisation de u_i, de telle manière que la matière première des produits rentre dans le système le plus tard possible. Dans ce cas, le niveau des encours est maintenu aussi faible que possible.

Tableau 3.4 – Date de début contrôlée de la première opération x_{i1k} de chaque produit

Jobs	J_1	J_2
u_i	1	6
x_{i1k}	1	6

Tableau 3.5 – Dates de fin C_i des produits

C_i	8	17
Avance	0	0

3.6 Ordonnancement des systèmes HVLV avec maintenance

Il est clair d'après la dynamique de la maintenance décrite par l'équation (2.12), que les activités de maintenance PM_{hk} ont été intégrées dans le modèle comme étant des opérations élémentaires d'un job. En effet, un problème de conflit entre les tâches de maintenance et les opérations O_{ijk} doit être résolu. Par conséquent, Les mêmes conditions utilisées dans la section 3.5.1, seront considérées sur les variables de décision $v_{ijk;hk}(T+1)$ et $v_{hk;zk}(T+1)$ afin de générer des ordonnancements faisables. Une politique de maintenance périodique est adoptée dans cette thèse. Dans ce cadre, des contraintes sur les dates de début x_{hk} des activités de maintenance seront intégrées dans le modèle afin de respecter les périodes entre deux tâches de maintenance successives. Un problème d'optimisation non-linéaire sous contraintes est résolu par la suite afin de minimiser le makespan et l'avance totale en juste-à-temps.

3.6.1 Génération d'un ordonnancement faisable

Selon les valeurs choisies des variables $v_{ijk;hk}(T+1)$ et $v_{hk;zk}(T+1)$, l'ordre d'exécution des tâches de maintenance PM_{hk} et des opérations O_{ijk} sur la machine M_k, sera de la manière suivante :

- Si $v_{ijk;hk}(T+1) = -\infty$, alors l'opération O_{ijk} est exécutée avant PM_{hk} sur la machine M_k pendant la période $T+1$. Par contre, si $v_{ijk;hk}(T+1) = 0$, alors c'est le cas inverse.

- Si $v_{hk;zk}(T+1) = -\infty$, alors PM_{hk} est exécutée avant PM_{zk} sur la machine M_k pendant la période $T+1$. Sinon, si $v_{hk;zk}(T+1) = 0$, alors c'est le cas inverse.

Les deux conditions citées ci-dessus sont choisies afin de résoudre le problème de conflit sur les machines. Pour chaque couple d'opérations concurrentes (O_{ijk}, PM_{hk}) et (PM_{hk}, PM_{zk}) ($h \neq z$), ces conditions peuvent être décrites par les relations (max,+) suivantes :

$$\begin{cases} v_{ijk;hk}(T+1) + v_{hk;ijk}(T+1) = -\infty \\ v_{hk;zk}(T+1) + v_{zk;hk}(T+1) = -\infty \end{cases} \quad \text{et} \quad \begin{cases} max(v_{ijk;hk}(T+1), v_{hk;ijk}(T+1)) = 0 \\ max(v_{hk;zk}(T+1), v_{zk;hk}(T+1)) = 0 \end{cases}$$

En utilisant les notations de l'algèbre (max,+), le système d'équation ci-dessus devient :

$$\begin{cases} v_{ijk;hk}(T+1) \otimes v_{hk;ijk}(T+1) = \epsilon \\ v_{hk;zk}(T+1) \otimes v_{zk;hk}(T+1) = \epsilon \end{cases} \quad \text{et} \quad \begin{cases} v_{ijk;hk}(T+1) \oplus v_{hk;ijk}(T+1) = e \\ v_{hk;zk}(T+1) \oplus v_{zk;hk}(T+1) = e \end{cases}$$

3.6.2 Mesure des performances de production : optimisation non linéaire sous contraintes

De la même manière que dans la section 3.5.2, nous considérons dans cette section une optimisation sur une seule période de production. Nous présentons dans cette section les contraintes d'optimisation dans le cas d'un ordonnancement avec maintenance et nous donnons par la suite quelques exemples illustratifs.

Minimisation du makespan

Les variables d'optimisation ici sont les variables de décision $v_{ijk;i''j''k}$, $v_{ijk;hk}$ et $v_{hk;zk}(\forall i, i'' = 1 \dots l, j, j'' = 1 \dots n, i \neq i'', j \neq j''$ et $z \neq h)$. Nous supposons que la matière première de chaque produit J_i est disponible à l'instant 0, c-à-d, $u_i = 0\ \forall i = 1 \dots l$ et que $t = 0$. Dans ce contexte, les contraintes de ce problème d'optimisation sont décrites par les équations suivantes :

$$x_{hk} = max(x_{ijk} + p_{ijk}; x_{zk} + t_{zk}; x_{ijk} + p_{ijk} + v_{hk;ijk}; x_{zk} + T_{(.)} + v_{hk;zk}) \tag{3.9}$$

$$x_{ijk} = max(x_{i'j'k} + p_{i'j'k}; x_{i''j''k} + p_{i''j''k} + v_{ijk;i''j''k}; u_i; x_{hk} + t_{hk} + v_{ijk;hk}) \tag{3.10}$$

$$x_{ijk} = max(x_{i'j'k} + p_{i'j'k}; x_{i(j-1)t} + p_{i(j-1)t}; x_{i''j''k} + p_{i''j''k} + v_{ijk;i''j''k}; x_{hk} + t_{hk} + v_{ijk;hk}) \tag{3.11}$$

$$\begin{cases} v_{ijk;i''j''k} + v_{i''j''k;ijk} = -\infty \\ v_{ijk;hk} + v_{hk;ijk} = -\infty \\ v_{hk;zk} + v_{zk;hk} = -\infty \end{cases} \quad \text{et} \quad \begin{cases} max(v_{ijk;i''j''k}, v_{i''j''k;ijk}) = 0 \\ max(v_{ijk;hk}, v_{hk;ijk}) = 0 \\ max(v_{hk;zk}, v_{zk;hk}) = 0 \end{cases} \tag{3.12}$$

Comme indiqué dans le chapitre 2, nous considérons dans ce chapitre deux types de maintenance périodique : maintenance répétitive périodique et maintenance flexible périodique. Deux cas sont distingués :

- **cas 1 :** les interventions de maintenance sont générées périodiquement sur une machine M_k avec des périodes égales T_k. Dans ce cas, il faut intégrer la période T_k dans

le modèle d'ordonnancement proposé (équation (2.11)) de telle sorte que l'intervalle de temps entre deux activités de maintenance successives soit égal à T_k. En plus, la première intervention de maintenance doit commencer à la date T_k sur la machine M_k et la dernière opération de maintenance doit finir à l'instant $\sum_{h=1}^{x} t_{hk} + T_k * x$. Une expression qui décrit cette contrainte peut être formalisée dans l'algèbre (max,+) comme suit :

$$T_k \leqslant x_{hk} \leqslant \sum_{h,z=1;h\neq z}^{x} t_{zk} + T_k * x \tag{3.13}$$

- **cas 2 :** les intervalles de temps, $T_{hk,zk}$ $(h, z = 1 \ldots x, h \neq z)$, entre deux opérations de maintenance successives PM_{hk} et PM_{zk} ne sont pas forcément égaux. Ils peuvent être différents mais ils sont donnés. La date de début T_{0k} de la première maintenance flexible sur la machine M_k est connue. Par conséquent, la période flexible entre deux activités de maintenance doit être respectée. Cette contrainte peut être décrite par la relation (max,+) suivante :

$$T_{0k} \leqslant x_{hk} \leqslant T_{0k} + \sum_{h,z=1;h\neq z}^{x} t_{zk} + max(\sum_{h,z=1;h\neq z}^{x} T_{hk,zk}) \tag{3.14}$$

Le terme $\sum_{h,z=1;h\neq z}^{x} T_{hk,zk}$ dans l'inégalité (3.14) est une addition de $(x-1)$ périodes $T_{hk,zk}$ (x est le nombre des opérations de maintenance flexible sur la machine M_k). Si $T_{0k} = T_{hk,zk} = T_k$ (les intervalles de temps entre les opérations de maintenance sont tous égaux), alors l'inégalité (3.14) se ramène à l'inégalité (3.13).

Exemple illustratif : L'algorithme d'optimisation décrit par l'équation (2.21) est appliqué afin de minimiser le makespan C_{max}.
L'affectation des activités de maintenance à chaque machine est comme suit :
- Deux activités de maintenance préventive répétitive périodique sur M_1 : PM_{11} et PM_{21}.
- Trois activités de maintenance préventive flexible périodique sur M_2 : PM_{12}, PM_{22} et PM_{32}.

Les durées t_{hk} allouées aux différentes activités de maintenance sont présentées dans le tableau 3.6. Nous supposons que la période $T_1 = 7$.

Tableau 3.6 – Durées des différentes activités de maintenance.

Durée	Valeur
t_{11}, t_{12}	2
t_{21}, t_{22}, t_{32}	3

Le tableau 3.7 présente les différentes valeurs de périodes $T_{hk,zk}$ des maintenances flexibles sur la machine M_2.

La valeur optimale obtenue du makespan est $C^*_{max} = 29$ unités de temps (voir figure 3.9).

Tableau 3.7 – Périodes $T_{hk,zk}$.

Période	T_{02}	$T_{12,22}$	$T_{12,32}$	$T_{22,32}$
Valeur	5	3	7	5

Le diagramme de Gantt présenté sur la figure 3.9 montre l'enchainement des opérations ainsi que les activités de maintenance et les dates de fin C_i des différents jobs $J_i(i = 1 \dots 2)$ récapitulées dans le tableau 3.8. La figure 3.9 montre que les

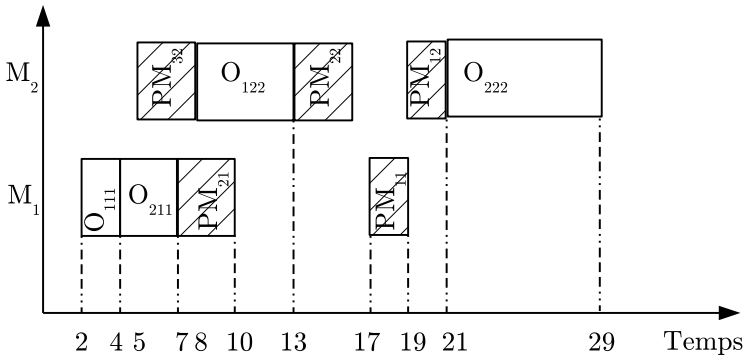

Figure 3.9 – Ordonnancement des opérations et des activités de maintenance sur les machines.

Tableau 3.8 – Dates de fin des produits

Jobs	J_1	J_2
C_i	13	**29**

résultats obtenus par le modèle (max,+) proposé (équation '2.11)) sont probants ce qui confirme la viabilité de notre modèle dans le cas des systèmes HVLV soumis à des maintenances préventives. En effet, les contraintes sur les maintenances intégrées dans le modèle représentées par les équations 3.13 et 3.14 sont respectées. Si on s'intéresse par exemple à la machine M_1, il est clair d'après le diagramme de Gantt (figure 3.9) que les intervalles de temps entre deux activités de maintenance répétitives périodiques successives sont égales et la période entre elles est telle que $T_1 = 7$ unités de temps. Par contre, la machine M_2 est soumise à des tâches de maintenance flexibles périodiques.

Par conséquent, les périodes entre les opérations de maintenance ne sont pas égales. Ceci est cohérent avec les résultats de simulation obtenus. En effet, d'après le diagramme de Gantt, la maintenance PM_{32} commence à l'instant $t = 5$. Cette valeur est bien égale à T_{02}. En plus, PM_{32} finit à $t = 8$. La date de début x_{22} de PM_{22} est égale à 13. Donc, l'intervalle de temps entre PM_{32} et PM_{22} est égale à 5. Cette valeur est bien celle de $T_{32,22}$ proposée dans le tableau 3.7. De même, les résultats obtenus montrent que la date de fin y_{22} de PM_{22} est égale à 16 et la date de début x_{12} de PM_{12} est égale à 19.

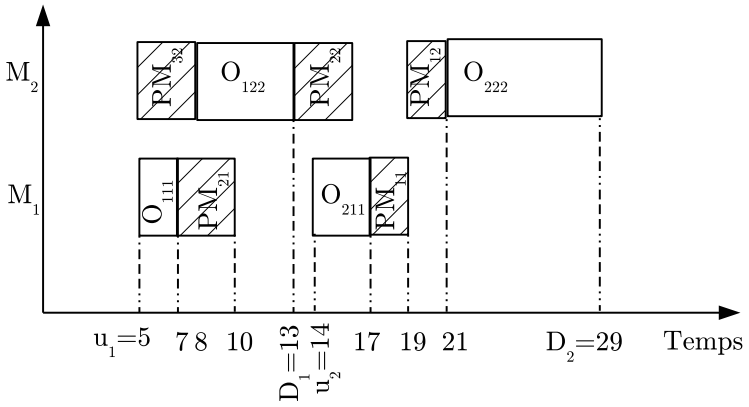

Figure 3.10 – Ordonnancement des opérations et des activités de maintenance sur les machines.

D'où, la période entre PM_{22} et PM_{12} est égale à $T_{22,12} = 3$. Par conséquent, les résultats présentés semblent intéressants et montrent que l'intégration de la maintenance dès la phase de conception de l'ordonnancement respecte la qualité des solutions construites. Les opérations sur les produits ainsi que les activités de maintenance sont ordonnées entre elles tel que le makespan est minimal.

Minimisation de l'avance totale en juste-à-temps

Dans le cas de minimisation de l'avance totale avec maintenance, les variables de contrôle sont : u_i et les mêmes variables de décision utilisées pour minimiser le makespan (section 3.6.2). Dans ce contexte, les contraintes de ce problème d'optimisation sont décrites par les équations (3.9), (3.10), (3.11), (3.12), (3.13), (3.14), (3.7) et (3.8).

Exemple illustratif : L'algorithme d'optimisation sous contrainte décrit par l'équation (2.22) est appliqué sur une seule période afin de minimiser l'avance totale R. Soit D_i les dates de besoin des jobs $J_i (i = 1 \dots 2)$ (Tableau 3.9).

Tableau 3.9 – Dates de besoin D_i des produits

Jobs	J_1	J_2
D_i	13	29

La valeur optimale obtenue de l'avance totale est $R = -19$ (figure 3.10). Remarquons que le critère R portant sur l'effort de la commande U est égale à -19. R représente la somme des commandes u_1 et u_2. Le diagramme de Gantt montre l'enchainement des opérations des différents jobs J_i $(i = 1 \dots 2)$. Le tableau 3.10 montre que le critère de la production en juste à temps est satisfait. En effet, les dates de début de la première opération x_{i1k} de chaque job J_i est inférieure ou égale à u_i la date d'entrée de chaque produit.

Tableau 3.10 – Date de début contrôlée de la première opération x_{i1k} de chaque produit

Jobs	J_1	J_2
u_i	5	14
x_{i1k}	5	14

Par conséquent, l'exécution des jobs J_1 et J_2 sur la machine M_1 commence le plus tard possible afin d'éviter les encours à l'entrée du système.

3.7 Conclusion

Ce chapitre est consacré à l'ordonnancement des systèmes de type HVLV sans et avec maintenance. La stratégie de prise de décision nécessite d'imposer des bornes sur les variables de contrôle afin de générer des ordonnancements faisables.
Des relations de coopération entre la production et la maintenance préventive sont mises en place dans les systèmes HVLV, à travers des relations dans l'algèbre (max,+) pour traiter le problème d'ordonnancement.
Les activités de maintenance ainsi que les opérations élémentaires sur les jobs sont ordonnancées simultanément entre elles afin de satisfaire un certain objectif. L'ordre des activités de maintenance sur les machines n'est pas connu à l'avance.
Pour résoudre le problème d'ordonnancement dans les systèmes HVLV, un problème d'ordonnancement non-linéaire sous contraintes est résolu afin de minimiser le makespan et l'avance totale en juste-à-temps. Les résultats obtenus par le modèle (max,+) sont probants ce qui confirme la viabilité de notre modèle.

Chapitre 4

Application de l'approche de pilotage

4.1 Introduction

L'objectif de ce chapitre est de valider la faisabilité de l'approche de pilotage (ordonnancement) développée dans les précédents chapitres et de tester la robustesse de notre approche. Les performances du pilotage sont évaluées à l'aide du logiciel d'optimisation non-linéaire appelé *LINGO*.
Dans la construction du modèle (max,+) le nombre des équations du modèle augmente d'une manière très importante quand la dimension du système est relativement importante. Par conséquent une génération automatique du modèle à l'aide d'un algorithme informatique sera de grande importance. La génération des équations est réalisée à l'aide d'un algorithme implanté sous *MATLAB*. Dans un premier temps, nous détaillons la logique de génération automatique du modèle d'ordonnancement proposé.
Dans un second temps, nous donnons l'extension de l'approche de pilotage aux cas de systèmes HVLV organisés en job-shop avec un seul cycle de production. Nous déclinerons et validerons notre approche sur deux exemples complexes plus proche de la réalité industrielle. Le premier est un job-shop carré (6×6) et le deuxième est un job-shop carré plus complexe (10×10) [12].
Dans la première partie de ce chapitre, l'approche de pilotage proposée est appliquée sans maintenance. Dans la deuxième partie, une stratégie de maintenance préventive systématique est considérée. Dans les deux cas, nous cherchons à minimiser deux critères couramment utilisés dans la littérature, à savoir : le makespan C_{max} et l'avance totale R en juste à temps.

4.2 Ordonnancement sans maintenance

Dans cette partie, des résultats de simulation en utilisant la méthodologie d'ordonnancement proposée sont présentés. La performance de la technique développée et évaluée à travers deux critères de performance, à savoir le "makespan" et l'avance totale. Les résultats obtenus sont commentés, analysés et comparés à ceux obtenus dans [12] où les auteurs exploitent un ordonnancement à l'aide d'une technique d'optimisation approchée (utilisation des algorithmes génétiques).

4.2.1 Logique de génération automatique des équations du modèle

Pour des raisons de simplicité de notation, nous considérons dans l'application de notre approche que les job shops sont carrés, c-à-d, $m = l = r$. Avec :
– m est le nombre des machines,
– l est le nombre des produits et
– r est le nombre des opérations sur chaque produit.
Par conséquent, d'une part, la taille du vecteur d'état X du modèle est égale à m^2. D'autre part, le nombre total des variables de décision $v_{ijk,i'j'k}$ dans la matrice de contrôle V appelé Q est égale à $(r-1)\times 2\times l$. Or $m = l = r$, alors $Q = (m-1)\times m\times 2$. Nous constatons que le nombre des variables du système explose rapidement si ce dernier est plus complexe. Par conséquent, le nombre des équations décrivant la dynamique du système augmente elle aussi d'une manière significative. Comme titre d'exemple pour un job shop de dimension 6×6, le nombre des contraintes est égale à 607. Alors que pour un job shop de taille 10×10, on a 2891 contraintes. Une génération automatique de ces équations est indispensable.
Nous donnons ici une approche de génération automatique des équations du modèle sans maintenance. Les données du système sont organisées dans un tableau « **Data** » comme suit :

1. La première colonne de « **Data** » contient l'indice des machines (k).
2. La deuxième colonne de « **Data** » contient l'indice des jobs (i).
3. La troisième colonne de « **Data** » contient l'indice des opérations (j).
4. La quatrième colonne de « **Data** » contient les temps opératoires p_{ijk}.

la taille de « **Data** » est égale à $m^2 \times 4$. Le tableau « **Data** » est comme suit (voir Tableau 4.1) :

Tableau 4.1 – La présentation des données du système

	indice machine	indice job	indice opération	temps opératoire
m^2 lignes	k	i	j	p_{ijk}
	–	–	–	–
	–	–	–	–

Chaque ligne de « **Data** » décrit l'information concernant une seule opération.
La méthodologie de conception du modèle, à partir de « **Data** », est illustrée avec l'algorithme présenté dans l'annexe 4.4 (voir algorithme 1). Toutes les équations générées sont sauvegardées dans un fichier nommé « **Sor** » . $y(m, n)$ représente l'élément de la matrice « **Data** » situé dans la ligne m et la colonne n.

4.2.2 Exemple 1

Le premier exemple [12] est pris d'un cas industriel. Il s'agit d'un système HVLV carré de dimension (6×6) (6 types de produits et 6 machines). Cet exemple permet de montrer les performances de la méthode proposée en présence d'une grande variété de produits et une utilisation concurente des machines (voir tableau 4.2).

Tableau 4.2 – Données de production de l'exemple 1

Job	numéro de machine/temps opératoire					
J_1	3/1	1/3	2/6	4/7	6/3	5/6
J_2	2/8	3/5	5/10	6/10	1/10	4/4
J_3	3/5	4/4	6/8	1/9	2/1	5/7
J_4	2/5	1/5	3/5	4/3	5/8	6/6
J_5	3/9	2/3	5/5	6/4	1/3	4/1
J_6	2/3	4/3	6/9	1/10	5/4	3/1

La dynamique du système du premier exemple est décrite par l'équation d'état (2.6).
où : X, U, Y, A, B, C et V ont la même signification que dans le modèle (2.6) décrite dans la section 2.3.2 ($n = 36$ et $l = 6$).
Dans le cas de minimisation du makespan, les contraintes d'optimisation sont décrites par les équations (3.3), (3.4), (3.5) et (3.6).
Dans le cas de minimisation de l'avance totale, les équations (3.7) et (3.8) s'ajoutent aux contraintes du problème d'optimisation du makespan.
où : $k, i, i'' = 1 \dots 6,\ j, j'' = 1 \dots 36,\ i \neq i''$ et $j \neq j''$.

Minimisation du makespan : Dans ce paragraphe, nous avons appliqué le problème d'optimisation non linéaire sous contraintes décrit par l'équation (2.21) avec $t = 0$ et $u_i = 0\ \forall 1 \leq i \leq l$ ($l = n = m$). La valeur optimale obtenue du makespan est $C^*_{max} = 55$ unités de temps (figure 4.1). Le diagramme de Gantt présenté à la figure 4.1 montre l'enchainement des opérations et les dates de fin C_i des différents jobs $J_i (i = 1 \dots 6)$ récapitulées dans le tableau 4.3.

Tableau 4.3 – Dates de fin des produits

Jobs	J_1	J_2	J_3	J_4	J_5	J_6
C_i	36	54	**55**	54	**55**	**55**

Une comparaison entre le résultat obtenu en utilisant le modèle d'ordonnancement

(max,+) et celui obtenu par la méthode des algorithmes génétiques améliorés (*Improved Adaptive Genetic Algorithm (IAGA)*) proposée dans la littérature [12], montre que les deux valeurs obtenues du makespan optimal sont égales (55 unités de temps). Dans le modèle d'ordonnancement (max,+) proposé, il n'y a pas de paramètres à régler et il montre une certaine souplesse dans son utilisation. Par contre, dans [12], les auteurs doivent régler plusieurs paramètres tels que la taille de la population initiale, les probabilités de croisement et de mutation afin de converger vers la bonne valeur du makespan optimal. Le choix des bons paramètres demeure le souci majeur lorsqu'on veut utiliser les algorithmes génétiques. Souvent il faut faire appel à une expérimentation importante pour trouver les valeurs des paramètres les plus performants. Donc, la méthode proposée dans ce chapitre nous paraît plus facile à appliquer.

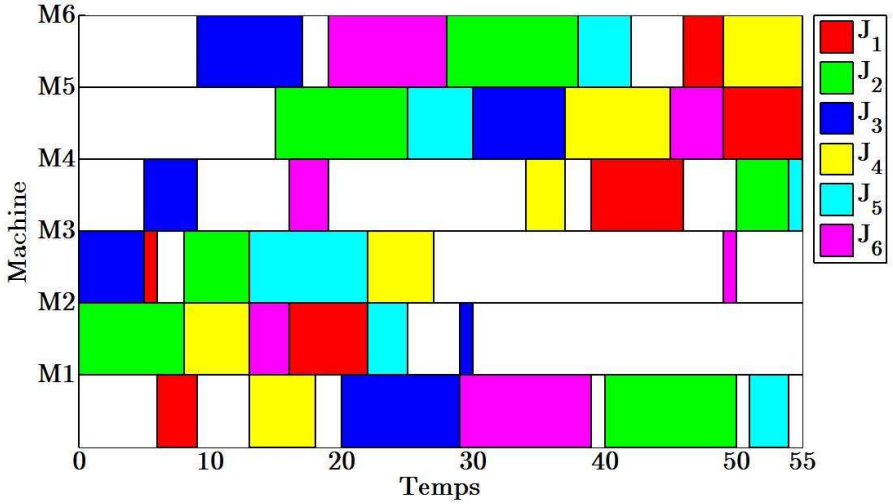

Figure 4.1 – Ordonnancement des opérations sur les machines

Il suffit juste de choisir un formalisme mathématique convenable qui décrit mieux le problème d'ordonnancement des systèmes HVLV à décision non libre ainsi que des valeurs des variables de décision adéquates pour générer des ordonnancements faisables sur les machines.

La minimisation du makespan est un critère qui favorise une utilisation élevée des ressources (machines). Dans ce contexte, soient $Toccup_k = \frac{\sum_{i,j,k=1}^{n} p_{ijk}}{C^*_{max}}$ le taux d'occupation des machines M_k et $T_{moy} = \frac{\sum_{k=1}^{n} Toccup_k}{n}$ le taux d'occupation moyen $(n = 1 \dots 6)$ (tableau 4.4).

Le tableau 4.4 montre que les machines M_1, M_5 et M_6 sont les plus occupées dans le système avec un taux d'occupation de 73%. Par contre, M_4 possède un taux d'occupation le plus petit (40%).

La valeur moyenne des taux d'occupation de toutes les machines (59%) montre que la valeur optimale obtenue du makespan par le modèle (max,+) proposé permet d'avoir une bonne occupation des ressources ($T_{moy} > 50\%$).

Tableau 4.4 – Taux d'occupation des machines et valeur moyenne

M_k	M_1	M_2	M_3	M_4	M_5	M_6
$Toccup_k$	0.73	0.47	0.47	0.4	0.73	0.73
T_{moy}	0.59					

Minimisation de l'avance totale en juste à temps : L'objectif de ce paragraphe est de minimiser la somme des avances R en juste à temps en utilisant le modèle (max,+) proposé dans la section. Pour ce faire, un problème d'optimisation non linéaire sous contraintes est résolu.
Soient $t = 0$ et D_i les dates de besoin des jobs $J_i (i = 1 \dots 6)$ (Tableau 4.5).

Tableau 4.5 – Dates de besoin D_i des produits

Jobs	J_1	J_2	J_3	J_4	J_5	J_6
D_i	60	77	55	36	45	55

La valeur optimale du critère $R^* = -80$ correspond à l'ordonnancement des opérations sur les machines représenté par le diagramme de Gantt dans la figure 4.2.

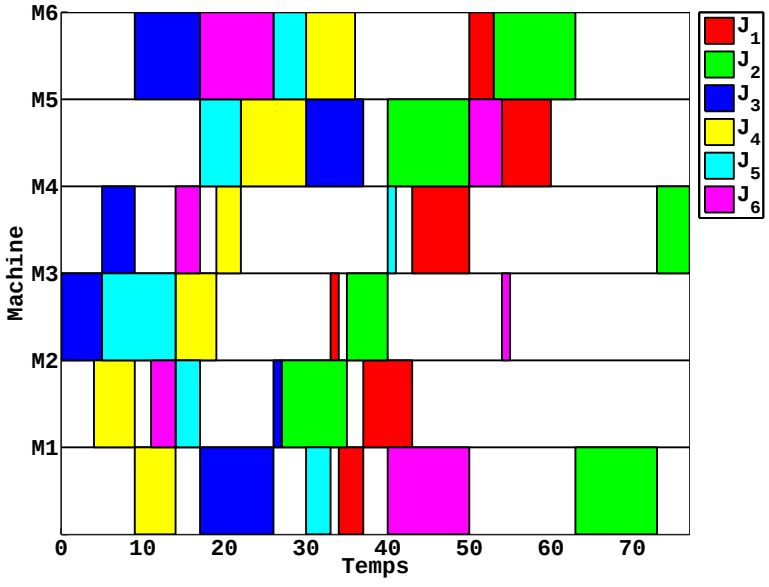

Figure 4.2 – Ordonnancement des opérations sur les machines

Le tableau 4.6 montre bien que les dates de besoin des produits sont respectées. En effet $\forall 1 \leq i \leq 6 : C_i \leqslant D_i$.

La production des jobs J_3 et J_5 est en avance, alors que la production de J_1, J_2, J_4 et J_6 est en juste à temps.

Tableau 4.6 – Dates de besoin D_i et dates de fin des produits C_i

Jobs	J_1	J_2	J_3	J_4	J_5	J_6
D_i	60	77	55	36	45	55
C_i	60	77	40	36	43	55
Avance	0	0	15	0	2	0

Le tableau 4.7 montre que le critère de la production en juste à temps est satisfait. En effet, les dates de début de la première opération x_{i1k} de chaque job J_i est inférieure ou égale à u_i la date d'entrée de la matière première de chaque produit dans le système. Nous constatons d'après le tableau 4.6, que tous les produits sont fabriqués à temps ($C_i = D_i$) sauf les produits P_3 et P_5 qui ont fini chacun avec un avance de 15 et 2 unités de temps. On peut conclure que les résultats d'optimisation satisfont bien le critère juste-à-temps de la production. Par conséquent la minimisation du critère proposé R entraîne une maximisation de u_i, de telle manière que la matière première des produits rentre dans le système le plus tard possible. D'où le niveau des encours est maintenu aussi faible que possible.

Tableau 4.7 – Date de début contrôlée de la première opération x_{i1k} de chaque produit

Jobs	J_1	J_2	J_3	J_4	J_5	J_6
u_i	33	27	0	4	5	11
x_{i1k}	33	27	0	4	5	11

4.2.3 Exemple 2

Le deuxième exemple est un système HVLV (10×10) plus complexe [12]. Le système correspondant a les caractéristiques d'un atelier job shop. Dix types de produits sont fabriqués sur dix machines. Chaque produit possède une gamme et un routage bien spécifique sur les machines. Il y a dix opérations par type de produit. D'où le vecteur d'état X dans l'équation (2.6) est de dimension 100×1. Le tableau 4.8 montre les différentes gammes des jobs J_i ainsi que les temps opératoires de chaque opération O_{ijk}.

La dynamique du système du second exemple est décrite par l'équation d'état (2.6) ($n = 100$ et $l = 10$).
Dans le cas de minimisation du makespan et de l'avance totale, les contraintes d'optimisation sont celles de l'exemple 1.
où : $k, i, i'' = 1 \ldots 10,\ j, j'' = 1 \ldots 100,\ i \neq i''$ et $j \neq j''$.

Tableau 4.8 – Données de production de l'exemple 2

Job	numéro de machine/temps opératoire									
J_1	1/6	2/3	5/2	3/8	7/3	6/3	4/4	9/8	8/2	10/6
J_2	3/1	6/5	4/3	10/8	8/6	2/11	7/2	1/3	9/5	5/9
J_3	2/11	1/3	9/7	3/2	10/10	7/2	8/3	6/4	5/4	4/7
J_4	4/3	6/5	8/12	7/5	1/3	9/3	2/8	3/4	5/12	10/9
J_5	7/2	6/6	10/10	9/6	5/1	1/1	2/2	3/8	4/3	8/7
J_6	3/3	8/9	9/4	4/6	6/4	7/2	1/2	2/3	5/2	10/9
J_7	3/3	7/1	4/12	9/4	10/2	1/8	2/3	5/2	6/8	8/7
J_8	3/3	4/5	10/4	8/3	6/4	5/10	2/5	7/2	1/3	9/11
J_9	10/8	7/1	4/12	9/4	3/2	8/8	2/3	6/8	5/8	1/7
J_{10}	3/6	10/5	4/4	8/3	6/4	5/10	9/5	7/6	1/3	2/11

La complexité de cet exemple réside dans le nombre très important des variables et des équations le décrivant. Les données générées par *LINGO* sont récapitulées sur le tableau 4.9. Ce tableau montre la différence de complexité entre les deux exemples :

Tableau 4.9 – Nombre des variables et des contraintes des deux exemples

–	Exemple 1	Exemple 2
Nombre total des variables	216	1000
Nombre des variables non linéaires	186	910
Nombre total des contraintes	607	2891
Nombre des contraintes non linéaire	181	901

Minimisation du makespan : Afin de prouver la viabilité du modèle d'ordonnancement proposé, nous allons comparer la valeur optimale du makespan obtenue par notre modèle avec celle obtenue en utilisant autres méthodes proposées dans la littérature.
La valeur optimale obtenue du makespan est $C^*_{max} = 85$ unités de temps (figure 4.3). Le diagramme de Gantt montré dans la figure 4.3 décrit l'enchainement des opérations et les dates de fin C_i des différents jobs $J_i (i = 1 \dots 6)$ récapitulées dans le tableau 4.10

Tableau 4.10 – Dates de fin des produits

Jobs	J_1	J_2	J_3	J_4	J_5	J_6	J_7	J_8	J_9	J_{10}
C_i	76	**85**	**85**	69	73	**85**	**85**	**85**	**85**	**85**

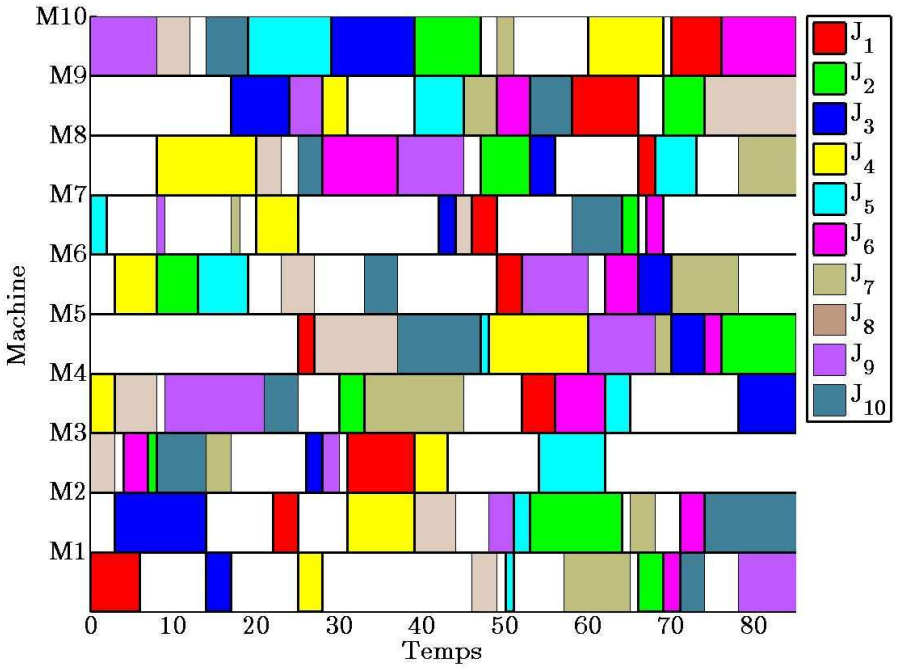

Figure 4.3 – Ordonnancement des opérations sur les machines

Les valeurs optimales du makespan obtenues en utilisant les algorithmes génétiques traditionnels et les algorithmes génétiques adaptatifs modifiés sont respectivement 114 et 94 unités de temps [11]. Alors que la valeur optimale obtenue par la méthode des algorithmes génétiques améliorés (*Improved Adaptive Genetic Algorithm (IAGA)*) proposée dans [12] est égale à 92 unités de temps. Ces résultats et le résultat obtenu par notre méthode. Cette constatation montre bien que la précision de convergence vers la solution optimale est meilleure en utilisant le modèle (max,+) proposé dans ce chapitre.

Minimisation de l'avance totale en juste à temps : L'objectif de ce paragraphe est de minimiser la somme des avances R en juste à temps. On considère la séquence des dates de besoin $D_i (i = 1 \dots 10)$ représentée sur le tableau 4.11. La valeur minimale trouvée de R correspond au Gantt représenté sur la figure 4.4.

Tableau 4.11 – Dates de besoin D_i et dates de fin de produits C_i

Jobs	J_1	J_2	J_3	J_4	J_5	J_6	J_7	J_8	J_9	J_{10}
D_i	**85**	87	86	90	89	**95**	**100**	**98**	**102**	**105**
C_i	**85**	82	83	79	87	**95**	**100**	**98**	**102**	**105**
Avance	**0**	5	3	11	2	**0**	**0**	**0**	**0**	**0**

D'après le tableau 4.11, on remarque que la commande optimale ($[U_{opt}, V_{opt}]$) calculée par l'algorithme d'optimisation respecte bien la spécification, c'est-à-dire que la sortie C_i est bien inférieure ou égale à la consigne D_i $\forall 1 \leq i \leq 10$.
Le tableau 4.12 montre que les encours sont minimisés. En effet, les premières opé-

rations de chaque produit commencent le plus tard possible ($x_{i1k} = u_i$). Par conséquent, une production en juste-à-temps est satisfaite.

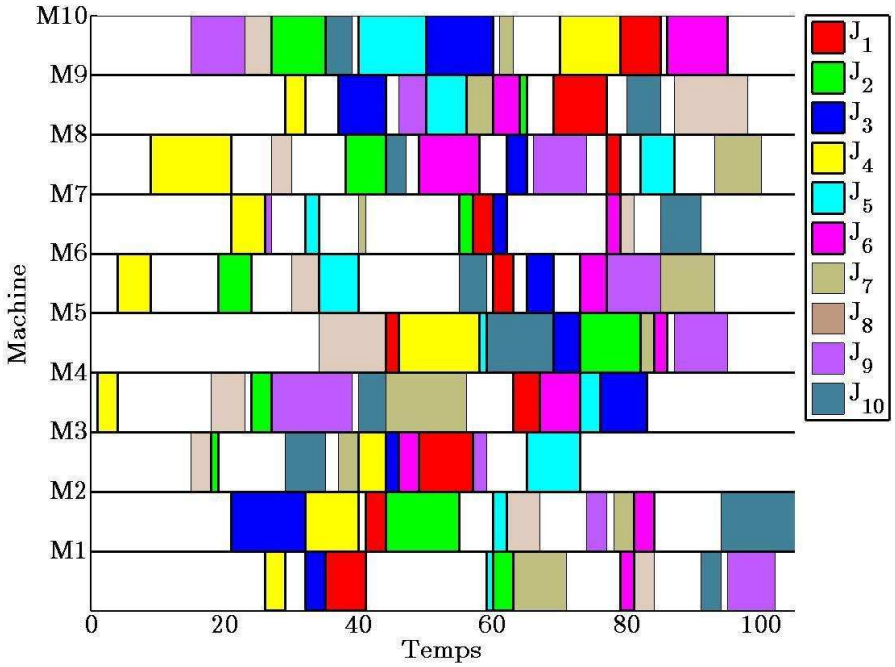

Figure 4.4 – Ordonnancement des opérations sur les machines

Tableau 4.12 – Date de début contrôlée de la première opération x_{i1k} de chaque produit

Jobs	J_1	J_2	J_3	J_4	J_5	J_6	J_7	J_8	J_9	J_{10}
u_i	35	18	21	1	32	46	37	15	15	29
x_{i1k}	35	18	21	1	32	46	37	15	15	29

4.3 Ordonnancement avec maintenance

Comme dans la section précédente, nous présentons dans cette section quelques résultats de simulation en utilisant la méthodologie d'ordonnancement proposée avec la présence de maintenance préventive. Deux exemples d'illustration sont considérés afin de mettre l'accent sur les points discutés dans les sections 2.3.2 et 3.6 et pour montrer les avantages des concepts proposés. Les mêmes indicateurs utilisés dans la section 4.2 sont utilisés de nouveau ici.

4.3.1 Logique de génération automatique des équations des activités de maintenance

Pour les même raisons citées dans la section 4.2.1 concernant la génération automatique des équations du modèle sans maintenance, l'augmentation du nombre des tâches de maintenance par machine rend l'implémentation du modèle très difficile.

Par conséquent, une approche pour la génération des équations des tâches de maintenance sera proposée.
Les données des activités de maintenance sont organisées dans un tableau « **MData** » comme suit :

1. La première colonne de « **MData** » contient l'indice des machines (k).
2. La deuxième colonne de « **MData** » contient l'indice h de l'opération de maintenance PM_{hk} ($1 \leqslant h \leqslant x$).
3. La troisième colonne de « **MData** » contient la durée t_{hk} de PM_{hk}.
4. La quatrième colonne de « **MData** » contient les valeurs des périodes des activités de maintenance.

La dimension de « **MData** » est égale à ($P \times 4$: P est le nombre total des opérations de maintenance). Le tableau « **MData** » est comme suit (Table 4.13) :

Tableau 4.13 – Présentation des données de maintenance

	indice machine	indice opération de maintenance	durée maintenance	période
	k	h	t_{hk}	$T_{(.)}$
P lignes	–	–	–	–
	–	–	–	–

En présence des maintenances, l'algorithme est présenté dans l'annexe 4.4 (voir algorithme 2). Il décrit la manière de générer les équations des activités de maintenance. Les tableaux « **Data** » et « **MData** » sont utilisés. Les équations générées sont sauvegardées dans le fichier de sortie « **Sor** ». $z(m, n)$ représente l'élément de la matrice « **MData** » situé dans la ligne m et la colonne n.

4.3.2 Application au premier exemple

Nous considérons dans cette section le cas d'un système HVLV de dimension (6×6) représenté par le tableau 4.2. Comme il est indiqué dans la section 2.3.2, nous considérons dans ce chapitre deux types de maintenance préventive (maintenance préventive répétitive périodique et maintenance préventive flexible périodique). L'affectation des activités de maintenance à chaque machine est comme suit :

1. Maintenance préventive répétitive périodique :
   - sur M_1 : PM_{11} et PM_{21}.
   - sur M_2 : PM_{12}, PM_{22} et PM_{32}.
   - sur M_5 : PM_{15}, PM_{25} et PM_{35}.
   - sur M_6 : PM_{16}, PM_{26} et PM_{36}.
2. Maintenance préventive flexible périodique :
   - sur M_3 : PM_{13}, PM_{23} et PM_{33}.
   - sur M_4 : PM_{14}, PM_{24} et PM_{34}.

Les durées t_{hk} allouées aux différentes activités de maintenance sont présentées dans le tableau 4.14.

Tableau 4.14 – Durées des différentes activités de maintenance.

Durée	Valeur
t_{11}, t_{23}, t_{33}, t_{36}	4
t_{21}, t_{14}, t_{34}, t_{16}, t_{26}	3
t_{12}, t_{15}, t_{25}	2
t_{22}	5
t_{32}	7
t_{13}	1
t_{24}, t_{35}	6

Le tableau 4.15 présente les différentes valeurs de périodes T_k des maintenances répétitives sur les machines correspondantes.

Tableau 4.15 – Périodes T_k entre activités de maintenance.

Période	T_1	T_2	T_5	T_6
Valeur	30	25	17	20

Le tableau 4.16 présente les différentes valeurs de périodes T_{0k} et $T_{hk,zk}$ des maintenances flexibles sur les machines M_3 et M_4.

Tableau 4.16 – Périodes T_{0k} et $T_{hk,zk}$.

Période	T_{03}	T_{04}	$T_{13,23}$	$T_{13,33}$	$T_{23,33}$	$T_{14,24}$	$T_{14,34}$	$T_{24,34}$
Valeur	15	20	7	5	6	8	9	15

Minimisation du makespan : Par application de la technique développée, La valeur optimale obtenue du makespan est $C^*_{max} = 67$ unités de temps (figure 4.5). Le diagramme de Gantt présenté à la figure 4.5 montre l'enchainement des opérations ainsi que les activités de maintenance et les dates de fin C_i des différents jobs $J_i (i = 1 \dots 6)$ récapitulées dans le tableau 4.17 ;

Tableau 4.17 – Dates de fin des produits

Jobs	J_1	J_2	J_3	J_4	J_5	J_6
C_i	**67**	59	36	**67**	63	60

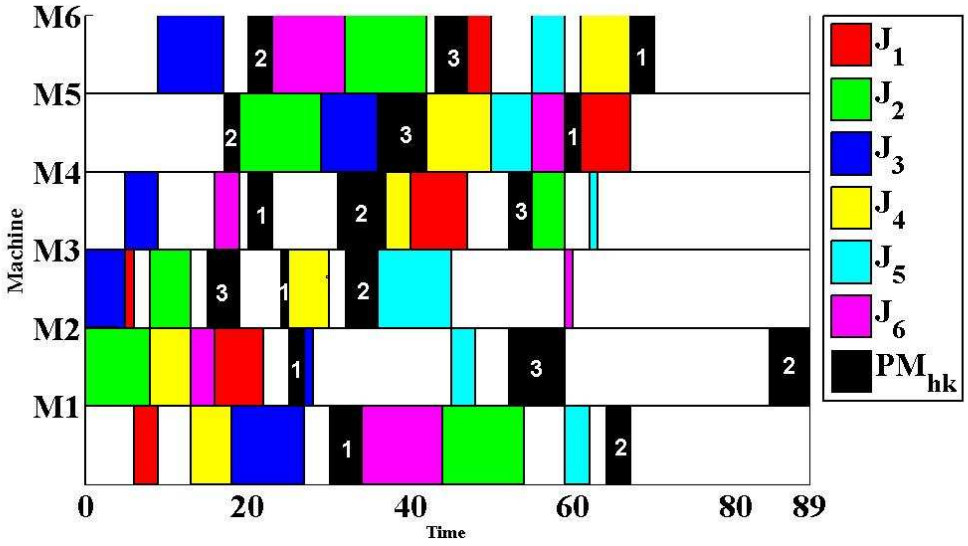

Figure 4.5 – Ordonnancement des opérations et des activités de maintenance sur les machines.

La figure 4.5 montre que les résultats obtenus par le modèle (max,+) proposé dans ce chapitre sont probants ce qui confirme la viabilité de notre modèle dans le cas des systèmes HVLV soumis à des maintenances préventives. En effet, les contraintes sur les maintenances intégrées dans le modèle représentées par les équations 3.13 et 3.14 sont respectées. Si on s'intéresse par exemple à la machine M_2, il est clair d'après le diagramme de Gantt (figure 4.5) que les intervalles de temps entre deux activités de maintenance répétitives périodiques successives sont égales et la période entre elles est telle que $T_2 = 25$ unités de temps. Par contre, la machine M_3 est soumise à des maintenances flexibles périodiques. Par conséquent, les périodes entre les opérations de maintenance ne sont pas égales. Ceci est cohérent avec les résultats de simulation obtenus. En effet, d'après le diagramme de Gantt, la maintenance PM_{33} commence à l'instant $t = 15$. Cette valeur est bien égale à T_{03}. En plus, PM_{33} finit à $t = 19$. La date de début x_{13} de PM_{13} est égale à 24. Donc, l'intervalle de temps entre PM_{33} et PM_{13} est égale à 5. Cette valeur est bien celle de $T_{13,33}$ proposée dans le tableau 4.16. De même, les résultats obtenus montrent que la date de fin y_{13} de PM_{13} est égale à 25 et la date de début x_{23} de PM_{23} est égale à 32. D'où, la période entre PM_{13} et PM_{23} est égale à $T_{13,23} = 7$. Par conséquent, les résultats présentés semblent intéressants et montrent que l'intégration de la maintenance dès la phase de conception de l'ordonnancement améliore la qualité des solutions construites. Les opérations des produits ainsi que les activités de maintenance sont ordonnées entre elles tel que le makespan est minimal.

La figure 4.5 montre que les machines sont inoccupées à partir de l'instant $t = 67$. Par conséquent, les activités de maintenance PM_{16} et PM_{22} peuvent être éliminées et planifiées dans le nouveau horizon de planification.

Minimisation de l'avance totale en juste à temps : L'objectif de ce paragraphe est de minimiser la somme des avances R en juste à temps en appliquant le problème d'optimisation non linéaire sous contraintes en présence de maintenance préventive. Pour ce faire, les variables $v_{ijk;i'j'k}, v_{ijk;hk}, v_{hk;zk}$ et les dates d'entrée de la matière première u_i des produits dans le système ($i = 1 \dots 6$) sont considérées comme des variables d'optimisation.
Soient $t = 0$ et D_i les dates de besoin des jobs $J_i (i = 1 \dots 6)$ (Tableau 4.18).

Tableau 4.18 – Dates de besoin D_i des produits

Jobs	J_1	J_2	J_3	J_4	J_5	J_6
D_i	73	67	70	69	74	76

La valeur optimale du critère $R^* = -110$ correspond à l'ordonnancement des opérations ainsi que les activités de maintenance sur les machines représenté par le diagramme de Gantt dans la figure 4.6. Le tableau 4.19 montre bien que les dates de besoin des produits sont respectées. En effet $\forall 1 \leq i \leq 6 : C_i \leqslant D_i$. La production des jobs J_3, J_4, J_5 et J_6 est en avance, alors que la production de J_1 et J_2 est en juste à temps.

Tableau 4.19 – Dates de besoin D_i et dates de fin des produits C_i

Jobs	J_1	J_2	J_3	J_4	J_5	J_6
D_i	**73**	**67**	70	69	74	76
C_i	**73**	**67**	40	60	71	66
Avance	**0**	**0**	30	9	3	10

Le tableau 4.20 montre que le critère de la production en juste à temps est satisfait. En effet, les dates de début de la première opération x_{i1k} de chaque job J_i est inférieure ou égale à u_i la date d'entrée de la matière première de chaque produit dans le système. Par conséquent la minimisation du critère proposé R entraîne une maximisation de u_i, de telle manière que la matière première des produits rentre dans le système le plus tard possible. D'où le niveau des encours est maintenu aussi faible que possible.

Tableau 4.20 – Date de début contrôlée de la première opération x_{i1k} de chaque produit

Jobs	J_1	J_2	J_3	J_4	J_5	J_6
u_i	46	1	3	9	37	14
x_{i1k}	46	1	3	9	37	14

La figure 4.6 montre que les machines sont inoccupées à partir de l'instant $t = 73$. Par conséquent, la maintenance PM_{12} peut être éliminée et planifiée dans le prochain horizon de planification.

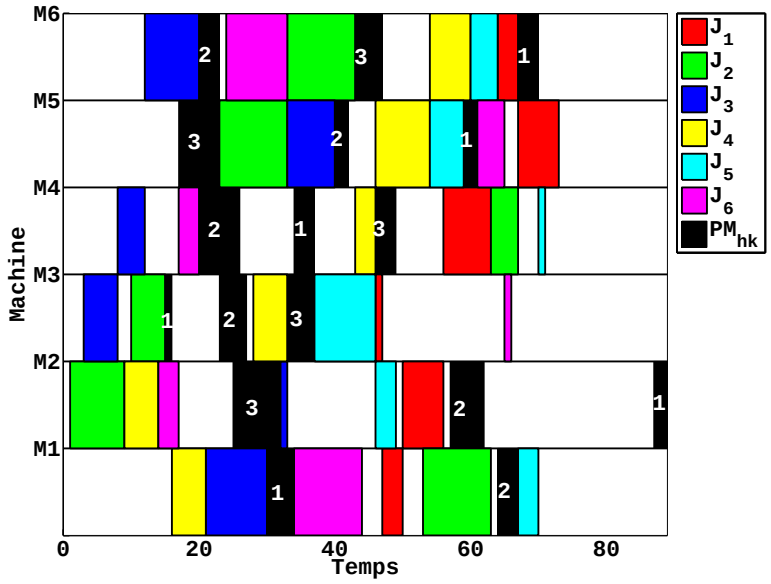

Figure 4.6 – Ordonnancement des opérations et des activités de maintenance sur les machines.

4.3.3 Application au second exemple

Nous considérons dans cette section l'exemple traité dans la section 4.2.3. C'est le cas d'un système HVLV de dimension (10×10). Nous supposons que ce système est soumis à des activités de maintenance préventive périodique. L'affectation des maintenances à chaque machine est comme suit :

1. Maintenance préventive répétitive périodique :
   - sur M_1 : PM_{11} et PM_{21}.
   - sur M_2 : PM_{12}, PM_{22} et PM_{32}.
   - sur M_5 : PM_{15}, PM_{25} et PM_{35}.
   - sur M_6 : PM_{16}, PM_{26} et PM_{36}.
   - sur M_7 : PM_{17}, PM_{27}, PM_{37} et PM_{47}.
   - sur M_8 : PM_{18} et PM_{28}.
   - sur M_9 : PM_{19}, PM_{29} et PM_{39}.
   - sur M_{10} : PM_{110}, PM_{210} et PM_{310}.
2. Maintenance Préventive Flexible Périodique :
   - sur M_3 : PM_{13}, PM_{23} et PM_{33}.
   - sur M_4 : PM_{14}, PM_{24} et PM_{34}.

Les durées t_{hk} allouées aux différentes activités de maintenance sont présentées dans le tableau 4.21.

Le tableau 4.22 présente les différentes valeurs des périodes T_k des maintenances répétitives sur les machines correspondantes.

Tableau 4.21 – Durées des différentes activités de maintenance.

Durée	Valeur
t_{11}, t_{23}, t_{33}, $t_{36}, t_{47}, t_{29}, t_{310}$	4
t_{21}, t_{14}, t_{34}, t_{16}, t_{26}, t_{27}, t_{28}	3
$t_{12}, t_{15}, t_{25}, t_{17}, t_{19}, t_{110}$	2
t_{22}, t_{18}	5
t_{32}	7
t_{13}, t_{37}	1
t_{24}, t_{35}, t_{210}	6
t_{39}	7

Tableau 4.22 – Périodes T_k.

Période	T_1	T_2	T_5	T_6	T_7	T_8	T_9	T_{10}
Valeur	30	25	17	20	15	10	19	40

Le tableau 4.23 présente les différentes valeurs des périodes T_{0k} et $T_{hk,zk}$ des maintenances flexibles sur les machines M_3 et M_4.

Nous avons appliqué le problème d'optimisation non linéaire sous contraintes avec

Tableau 4.23 – Périodes T_{0k} et $T_{hk,zk}$.

Période	T_{03}	T_{04}	$T_{13,23}$	$T_{13,33}$	$T_{23,33}$	$T_{14,24}$	$T_{14,34}$	$T_{24,34}$
Valeur	15	20	7	5	6	8	9	15

maintenance avec $t = 0$ et $u_i = 0$ $\forall 1 \leq i \leq 10$. La valeur optimale obtenue du makespan est $C^*_{max} = 129$ unités de temps (figure 4.7). Le diagramme de Gantt présenté sur la figure 4.7 montre l'enchainement des opérations ainsi que les activités de maintenance et les dates de fin C_i des différents jobs $J_i (i = 1 \ldots 10)$ récapitulées dans le tableau 4.24

Tableau 4.24 – Dates de fin des produits

Jobs	J_1	J_2	J_3	J_4	J_5	J_6	J_7	J_8	J_9	J_{10}
C_i	100	**129**	127	125	122	116	**129**	**129**	115	115

Les contraintes sur les maintenances intégrées dans le modèle représentées par les équations (3.13) et (3.14) sont respectées. Les opérations sur les produits ainsi que les activités de maintenance sont ordonnées simultanément entre elles tel que le makespan est minimal. Dans ce contexte, la machine M_7 est soumise à 4 interventions de maintenance répétitives périodiques. Il est clair d'après le diagramme de Gantt (figure 4.7) que les intervalles de temps entre deux activités de maintenance répétitives périodiques successives sont égales et la période entre elles est telle que $T_7 = 15$ unités de temps.

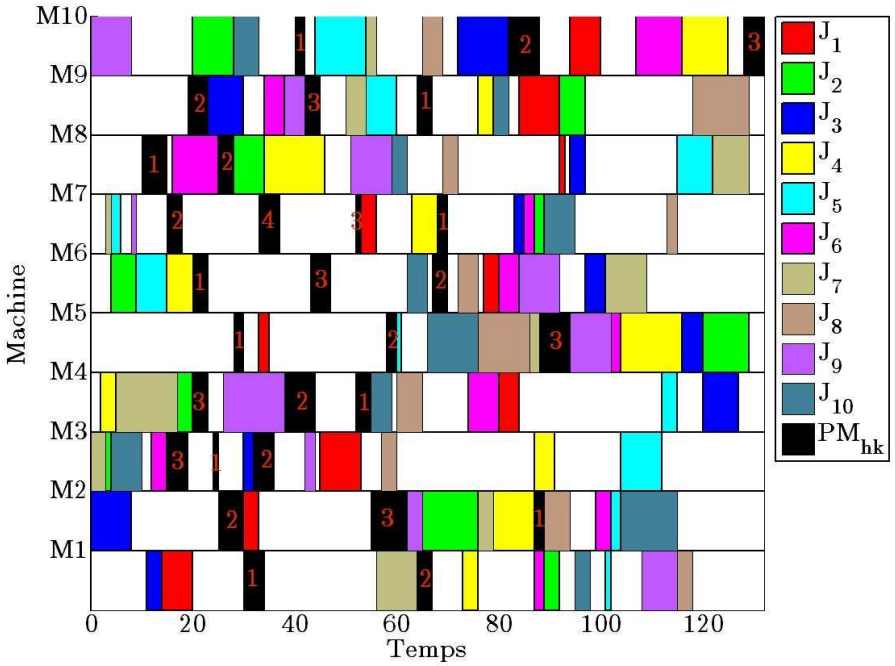

Figure 4.7 – Ordonnancement des opérations et des maintenances sur les machines

Par contre, la machine M_4 est soumise à des maintenances flexibles périodiques. Par conséquent, les périodes entre les opérations de maintenance ne sont pas égales. Ceci est cohérent avec les résultats de simulation obtenus. En effet, d'après le diagramme de Gantt, la maintenance PM_{34} commence à l'instant $t = 20$. Cette valeur est bien égale à T_{04}. En plus, PM_{34} finit à $t = 23$. La date de début x_{24} de PM_{24} est égale à 38. Donc, l'intervalle de temps entre PM_{34} et PM_{24} est égale à 15. Cette valeur est bien celle de $T_{34,24}$ proposée dans le tableau 4.23. De même, les résultats obtenus montrent que la date de fin y_{24} de PM_{24} est égale à 44 et la date de début x_{14} de PM_{14} est égale à 52. D'où, la période entre PM_{24} et PM_{14} est égale à $T_{14,24} = 8$. Par conséquent, les résultats présentés semblent intéressants et montrent que l'intégration de la maintenance dès la phase de conception de l'ordonnancement améliore la qualité des solutions construites. Les opérations sur les produits ainsi que les activités de maintenance sont ordonnées entre elles tel que le makespan est minimal.

4.4 Conclusion

A travers ce chapitre, nous avons illustré l'approche de pilotage (ordonnancement) des systèmes HVLV développée dans cette thèse afin de valider sa faisabilité sur des cas complexes. Ainsi, nous avons détaillé dans un premier temps, l'extension de l'approche de pilotage et aux cas de systèmes HVLV organisé en job-shop carré de dimension (6×6).

Dans un second temps, un deuxième exemple plus complexe, tiré de la littérature

(job-shop de dimension 10×10) [12], nous a servi pour confirmer la viabilité de notre méthodologie d'ordonnancement par rapport à d'autres techniques développé [11,12]. Aussi, la robustesse de l'approche vis-à-vis des aléas de fonctionnement du système et de son environnement (le cas de maintenance préventive) a pu être évaluée. Deux critères couramment utilisés ont été minimisés (le makespan et l'avance totale en juste à temps) dans les deux cas : sans et avec maintenance.

Conclusions et perspectives

Conclusions :

La complexité sans cesse croissante des systèmes de production et plus particulièrement les systèmes de type HVLV (*High-Variety, Low-Volume*) a fait des méthodes de modélisation et d'optimisation un des outils les plus répandus dont on dispose pour l'étude de ces systèmes à un coût raisonnable. Pour cette raison, toute technique visant à améliorer l'efficacité de cette approche est d'un très grand intérêt.

Les travaux de cette thèse portent sur la modélisation et l'optimisation des systèmes HVLV en utilisant l'algèbre (max, +). Dans ce contexte, nos travaux ont ciblé trois objectifs majeurs :

1. la proposition d'un cadre de modélisation adéquat afin de piloter les flux de production à faible cadence et à grande variété ;
2. le développement d'une méthodologie de synthèse d'un système de pilotage (système d'ordonnancement) dédié à une production de type HVLV et capable de réguler les flux de production au niveau physique (opérationnel) et d'assurer le respect et le suivi des spécifications globales (au niveau décisionnel) en tenant compte d'un attribut très important dans les systèmes HVLV au sens de la production à la demande (date de besoin) ;
3. la conception et la formalisation d'un système de pilotage (système d'ordonnancement) pour les systèmes HVLV en tenant compte de la maintenance préventive périodique.

Suite à une introduction générale, nous avons mené dans le premier chapitre une étude bibliographique portant sur la modélisation des systèmes de production. Une classification des SdPs ainsi qu'une caractérisation détaillée des systèmes HVLV nous a permis de choisir un outil de représentation adéquat pour ce type de systèmes. Nous avons vu dans ce chapitre que les systèmes HVLV sont des systèmes :

– à flux discret : dans ce genre de systèmes, les produits sont fabriqués en discontinu. Les quantités sont généralement restreintes et les produits variés. Dans ce cas, il s'agit de considérer individuellement chaque produit au sein du système ce qui permet de réaliser une traçabilité par l'attribut « **date de besoin** ». Cette carac-

téristique de systèmes HVLV nous a conduit sur une production à la commande. Au sein des systèmes HVLV, les responsables de planification ne considèrent que des produits à partir de demandes clients personnalisées dues à la variété de flux de production.
– qui peuvent avoir la structure de flow-shop généralisés ou des job-shop puisqu'ils présentent une diversité importante des flux de produits avec des séquences variées de production.

Après avoir présenté, au premier chapitre les caractéristiques des systèmes HVLV, nous avons fait un état de l'art sur les différents types et formalismes de modélisation des systèmes à évènements discret(SEDs). La littérature dispose de deux grandes catégories de représentation pour les SEDs, à savoir les représentations qualitatives et quantitatives. Nous nous sommes intéressés à la modélisation quantitative des systèmes HVLV plus précisément la modélisation en utilisant l'algèbre (max, +). Notre choix a été justifié par le fait que ce type de formalisme répond à des questions quantitatives et de performance (débit de système, encours, date de besoin, temps de séjour moyen, etc.). En plus, les modèles algébriques tel que l'algèbre (max, +) visent à capter la description des trajectoires des SEDs, y compris les systèmes HVLV en termes d'un ensemble d'opérations en fonction de l'état ou/et des événements d'une manière analogue à celle employée pour décrire les systèmes continus par les équations différentielles [28].

Au chapitre 2, nous avons adopté en premier temps un modèle (max, +) linéaire dans le sens de l'algèbre des dioïdes afin de représenter le comportement discret des systèmes HVLV à décision libre. Cette modélisation a permis d'expliciter, sous forme de représentation d'état dans l'algèbre (max, +), la relation entre les variables d'entrée (dates d'entrée de la matière première de produits dans le système), les variables de sortie (date de fin des opérations sur les produits) et les variables d'état (date de début des opérations pour chaque produit). A travers cette représentation, on pouvait dès lors transposer et appliquer des concepts de commande issus de l'automatique des systèmes continus pour traiter des problèmes de suivi de trajectoire de production (satisfaire les dates de besoin). Il existe dans la littérature deux grandes démarches pour la poursuite de trajectoire des systèmes (max, +) linéaires :

1. la poursuite de trajectoire en boucle ouverte. Ce type de commande est basé sur la théorie de **résiduation** pour l'inversion du système [9].
2. le suivi de trajectoire en boucle fermée. Il existe deux techniques de commande en boucle fermée :
   – la commande par modèle interne : elle sert à compenser l'erreur de modélisation qui peut exister dans le cas réel entre le système et son modèle de représentation en contrôlant le système par la différence entre la référence à suivre et l'erreur entre le modèle et le système.

– la démarche prédictive : elle est associée à un algorithme d'optimisation convexe sous contraintes et elle est appliquée pour résoudre un problème de poursuite de trajectoire en juste-à-temps.

Une extension des applications de l'algèbre (max, +) pour l'ordonnancement des systèmes HVLV à décision non libre sans et avec maintenance a été proposée en second temps. Dans ce cadre, nous avons supposé que les opérations à réaliser sont strictement non-préemptives, ce qui signifie qu'une fois commencée, l'exécution d'une opération ne pouvait être interrompue ni par la réalisation d'une tâche de maintenance, ni par celle d'une autre opération. Cette recherche présente une technique de prise de décision dans l'algèbre (max, +) pour l'ordonnancement des systèmes HVLV non libres. En effet, nous avons introduit des variables de décision (de contrôle) dans le modèle (max, +) proposé afin de résoudre le conflit entre les opérations réalisées sur les produits et les activités de maintenance nécessitant un traitement sur la même machine. Dans ce modèle, les dates de début des opérations et des tâches de maintenance sur les machines sont considérées comme des variables d'état du système. Les variables de contrôle (variables d'entrée) déterminent l'ordre des opérations sur les machines et par la suite les dates de fin des produits et des activités de maintenance (variables de sortie du système). Nous avons vu que le modèle proposé est non-linéaire dans le sens de l'algèbre (max, +) dû à la multiplication (max, +) entre les variables d'état et les variables de contrôle du système.

Dans le troisième chapitre, nous avons abordé la problématique d'ordonnancement des systèmes HVLV avec et sans maintenance. Après avoir donné quelques notions générales sur l'ordonnancement et après avoir ressitué les différentes méthodes d'ordonnancement des systèmes HVLV existantes dans la littérature, nous avons adopté une méthode d'ordonnancement exacte. Cette méthode consiste à établir des contraintes et des conditions sur les variables de décision du modèle afin de générer des ordonnancements faisables. Cette dernière est basée sur une approche d'ordonnancement non-linéaire sous contraintes. Elle vise à minimiser deux critères de performance couramment utilisés dans la littérature : le makespan et la date de besoin en juste-à-temps qui se traduit par la minimisation de la somme des avances. Une politique de maintenance préventive périodique a été adoptée. Deux cas d'étude sont considérés : le cas de la maintenance préventive répétitive périodique (périodes d'intervention égales) et le cas de la maintenance préventive flexible périodique (périodes d'intervention différentes). Dans ce cadre des contraintes sur les dates de début des opérations de maintenance ont été établies dans l'algèbre (max,+) afin de respecter les périodicités entre les tâches de maintenance. Des exemples académiques illustratifs pris dans la littérature ont été présentés dans ce chapitre afin de montrer la viabilité de notre approche.

Dans le quatrième et dernier chapitre de cette thèse, l'accent a été mis sur la faisabilité de la démarche de pilotage proposée sur des cas applicatifs complexes à grande variété de produits. Nous avons appliqué notre approche sur deux exemples complexes plus proche de la réalité industrielle. Le premier est un job-shop carré (6×6) et le deuxième est un job-shop carré plus complexe (10×10). Les résultats présentés semblent intéressants et montrent que l'intégration de la maintenance dès la phase de conception de l'ordonnancement ne remet pas en cause la qualité des solutions construites.

Perspectives :

Suite à ce travail de recherche, nous proposons dans cette section les perspectives suivantes :

- les systèmes HVLV sont caractérisés par une grande variété de produits. Ceci implique des changements de série sur les machines pour chaque type de produit. Le modèle proposé peut être amélioré afin de tenir compte des temps de changement de série (*set-up times*).
- le contexte actuel impose aux entreprises, y compris les systèmes HVLV des délais de plus en plus serrés, alors que la demande de plus en plus irrégulière ne permet plus d'anticiper la production. Dans cet environnement incertain, il est important d'organiser le travail prévu sur les ressources disponibles, d'intégrer au mieux dans cette organisation les commandes imprévues et urgentes, ainsi que de maîtriser les conséquences des aléas internes sur des programmes de fabrication de plus en plus tendus. Dans ce cadre, des méthodologies et des techniques peuvent être mises en œuvre pour résoudre les problèmes du type HVLV avec des demandes prévisionnelles ou incertaines. Ceci permet de mettre en cause l'ordonnancement à chaque fois qu'une commande urgente arrive dans le système.
- le cas des problèmes d'ordonnancement statique reste loin de la réalité industrielle. En effet, l'ordonnancement précis des tâches de maintenance et de production déterminé dans ce cadre est inévitablement perturbé par des aléas de production. En particulier, une politique de maintenance préventive, même très efficace, limite le risque de défaillances mais ne les élimine pas complètement. Il est donc parfois nécessaire de réagir à une panne en exécutant une opération de maintenance corrective le plus tôt possible. Dans ce contexte, le modèle proposé dans ce travail de recherche peut être étendu au cas de la maintenance corrective. Cet objectif sera réalisé en considérant des tâches de maintenance corrective qui apparaissent aléatoirement lors de l'ordonnancement des tâches de production et de maintenance systématique.
- le modèle (max, +) proposé est non linéaire. Dans ce cadre, des techniques de résolution des modèles (max, +) peuvent être développées directement dans l'algèbre des dioïdes sans être obligé de passer par l'optimisation classique.

Annexe

Données : Chargement des données du modèle depuis « **Data** »;
Sorties : Ouvrir un fichier de sortie « **Sor** »;
tantque $i \leqslant m^2$ **répéter**
  $i \leftarrow i + 1$;
  tantque $j \leqslant m^2$ **répéter**
    $j \leftarrow j + 1$;
    si $y(i,3) = 1$ **alors**
      Écrire dans « **Sor** » les relations :

$$x_{y(i,2)y(i,3)y(i,1)} \geqslant u_{y(i,2)}$$

$$x_{y(i,2)y(i,3)y(i,1)} \geqslant t$$

    finsi
    si $y(i,2) = y(j,2)$ *et* $y(i,1) \neq y(j,1)$ *et* $(y(i,3) = y(j,3) - 1$ *ou* $y(i,3) = y(j,3) + 1)$ **alors**
      Écrire dans « **Sor** » la relation :

$$x_{y(i,2)y(i,3)y(i,1)} \geqslant x_{y(j,2)y(j,3)y(j,1)} + y(j,4)$$

    sinon
      Écrire dans « **Sor** » les relations :

$$x_{y(i,2)y(i,3)y(i,1)} \geqslant x_{y(j,2)y(j,3)y(j,1)} + y(j,4) + v_{y(i,2)y(i,3)y(i,1);y(j,2)y(j,3)y(j,1)}$$

$$max(v_{y(i,2)y(i,3)y(i,1);y(j,2)y(j,3)y(j,1)}; v_{y(j,2)y(j,3)y(j,1);y(i,2)y(i,3)y(i,1)}) = 0$$

$$v_{y(i,2)y(i,3)y(i,1);y(j,2)y(j,3)y(j,1)} + v_{y(j,2)y(j,3)y(j,1);y(i,2)y(i,3)y(i,1)} = -\infty$$

    finsi
  fintantque
fintantque

Algorithme 1: Génération des équations du modèle

Données : Chargement des données du modèle depuis « **Data** »;
Chargement des données des tâches de maintenance depuis « **MData** »;
Sorties : Ouvrir un fichier de sortie « **Sor** »;
Initialisation : $i \leftarrow 1$;
pour j *allant de* 1 *à* P **faire**
  pour k *allant de* 1 *à* $m1$ **faire**
    pour l *allant de* 1 *à* $m1$ **faire**
      Trouver les numéros des lignes de $z(:,1)$ tel que $z(:,1) = P$ et les regrouper dans un vecteur colonne **« o »** : $o = find(z(:,1) == j)$;
      Calculer la dimension **« [m1 n1] »** du vecteur **« o »**;
      Initialisation : $total(i) \leftarrow 0$;
      $total(i) = total(i) + z(o(l), 3)$;
      $total(i) = total(i) - z(o(k), 3) + m1 * z(o(l), 4)$;
      $i \leftarrow i + 1$;
    finpour
  finpour
finpour
pour i *allant de* 1 *à* m^2 **faire**
  pour j *allant de* 1 *à* P **faire**
    pour k *allant de* 1 *à* P **faire**
      si $y(i,1) = z(j,1)$ **alors**
        Écrire dans **Sor** les relations :

$$z(j,4) \leqslant x_{z(j,2),z(j,1)} \leqslant total(j)$$

$$x_{y(i,2),y(i,3),y(i,1)} \geqslant x_{z(j,2),z(j,1)} + z(j,3) + v_{y(i,2),y(i,3),y(i,1);z(j,2),z(j,1)}$$

$$x_{z(j,2),z(j,1)} \geqslant x_{y(i,2),y(i,3),y(i,1)} + y(i,4) + v_{z(j,2),z(j,1);y(i,2),y(i,3),y(i,1)}$$

$$max(v_{y(i,2),y(i,3),y(i,1);z(j,2),z(j,1)}; v_{z(j,2),z(j,1);y(i,2),y(i,3),y(i,1)}) = 0$$

$$v_{y(i,2),y(i,3),y(i,1);z(j,2),z(j,1)} + v_{z(j,2),z(j,1);y(i,2),y(i,3),y(i,1)} = -\infty$$

      finsi
      si $z(j,1) = z(k,1)$ *et* $j \neq k$ **alors**
        Écrire dans **Sor** les relations :

$$x_{z(j,2),z(k,1)} \geqslant x_{z(j,2),z(j,1)} + z(j,3) + z(j,4) + v_{z(k,2),z(k,1);z(j,2),z(j,1)}$$

$$max(v_{z(k,2),z(k,1);z(j,2),z(j,1)}; v_{z(j,2),z(j,1);z(k,2),z(k,1)}) = 0$$

$$v_{z(k,2),z(k,1);z(j,2),z(j,1)} + v_{z(j,2),z(j,1);z(k,2),z(k,1)} = -\infty$$

      finsi
    finpour
  finpour
finpour

Algorithme 2: Génération des contraintes des activités de maintenance

Bibliographie

[1] H. Pierreval, R. Bruniaux, and C. Caux. A continuous simulation approach for supply chains in the automotive industry. *Simulation Modelling Practice and Theory*, 15 :185–198, 2007.

[2] K. Tamani. *Développement d'une méthodologie de pilotage intelligent par régulation de flux adaptée aux systèmes de production.* PhD thesis, Université de Savoie, 2008.

[3] J.E. Hopcroft, R. Motwani, and J.D. Ullman. *Introduction to automata theory, languages, and computation*, volume 3. Addison-Wesley Reading, MA, 1979.

[4] T. Murata. Petri nets : properties, analysis and applications. *Proc. IEEE*, 77, N4 :541–580, avril 1989.

[5] H. T. Papadopoulos, C. Heavey, and J. Browne. *Queueing Theory in Manufacturing Systems Analysis and Design.* Chapman & Hall, 1993.

[6] F.L. Baccelli, G. Cohen, G.J. Olsder, and J.P. Quadrat. *Synchronization and linearity.* Wiley New York, 1992.

[7] T. Yamada and R. Nakano. Genetic algorithms for job-shop scheduling problems. Technical report, Modern Heuristic for Decision Support, London, England, 1997.

[8] W. L. Wang, Q. D. Wu, and Y. Song. Modified adaptive genetic algorithms for solving job-shop scheduling problems. *Systems Engineering-Theory and Practice*, 2 :58–62, 2004.

[9] L. Wang and D.B. Tang. An improved adaptive genetic algorithm based on hormone modulation mechanism for job-shop scheduling problem. *Expert Systems with Applications*, 38 :7243–7250, 2011.

[10] A. Banerjee and J. S. Burton. Equipement utilization based maintenance task scheduling in a job shop. *European Journal of Operational Research*, 45 :191202, 1990.

[11] Y. Harrath. *Contribution à l'ordonnancement conjoint de la production et de la maintenance : application au cas d'un job shop.* PhD thesis, Université de Franche-Comté, 2003.

[12] L. Houssin. Contribution à l'étude des systèmes à évènements discrets dans l'algèbre des dioïdes. applications aux systèmes de transport. Technical report, Laboratoire d'Automatique et Informatique Appliquée : Nantes-Angers, 2003.

[13] J.-Y. Le Boudec and P. Thiran. *Newtork Calculus.* Springer-Verlag, 2001.

[14] G. Draghici, N. Brinzei, and I. Filipas. La modélisation et la simulation en vue de la conduite des systèmes de production. In *Les cahiers des Enseignements Francophones en Roumanie, Bucarest*, 1998.

[15] H. Dhouibi. *Utilisation des réseaux de petri à intervalles pour la régulation d'une qualité : application à une manufacture de tabac.* PhD thesis, Université des Sciences et Technologies de Lille, 2005.

[16] G. Rodde. *Les systèmes de production - Modélisation et performances.* Hermès, 1989.

[17] D. Trenteseaux and O. Sénéchal. Conduite des systèmes de production manufacturière. Technical report, Technique de l'ingénieur, S 7 598, 2002.

[18] A. M'Halla. *Contribution à la gestion des perturbations dans les systèmes manufacturiers à contraintes de temps.* PhD thesis, Ecole Centrale de Lille, Ecole Nationale d'Ingénieurs de Tunis, 2010.

[19] V. Giard. *Gestion de la production et des flux.* Economica, 2003.

[20] D. Andreu. *Commande et supervision des procédés discontinus : une approche hybride.* PhD thesis, Université Paul Sabatier de Toulouse, 1996.

[21] P. Baranger. *Gestion de la production.* Collection vuibert entreprise, 1987.

[22] H. Huang and S.A Irani. An enhanced systematic layout planning process for high-variety low-volume (HVLV) manufacturing facilities. In *The 17th International Conference on Production Research*, 2003.

[23] J.P. Meinadier. *Ingénierie et intégration des systèmes. Études et logiciels informatiques.* Hermès, Paris, 1998.

[24] K. Tamani, R. Boukezzoula, and G. Habchi. Intelligent distributed and supervised flow control methodology for production systems. *Engineering Applications of Artificial Intelligence*, 22 :1104–1116, 2009.

[25] K. Tamani, R. Boukezzoula, and G. Habchi. Multi-objective supervisory flow control based on fuzzy interval arithmetic : Application for scheduling of manufacturing systems. *Simulation Modelling Practice and Theory*, 19(5) :1371 – 1383, 2011.

[26] Y.C. Ho. Dynamics of discrete event systems. *Proceedings of the IEEE*, 77, No. 1 :3–6, 1989.

[27] X.R Cao and Y.C Ho. Models for discrete event dynamic systems. *IEEE Control Systems Magazine*, 10 :69–76, 1990.

[28] J.P. Calvez. *Spécification et conception des systèmes.* Masson, 1990.

[29] G.W. Brams. *Réseaux de Petri : théorie et pratique.* Masson, 1983.

[30] R. David and H. Alla. *Du Grafcet au réseaux de Petri.* Hermès, 1989.

[31] A. Alla, R. David, M. Di Mascolo, and J.-L. Ferrier. *Analyse et commande des systèmes à évènements discrets.* Hermès, 2000.

[32] A. Arnold. *Systèmes de transitions et sémantique des processus communicants.* Masson, 1992.

[33] B. Cottenceau. *Contribution à la commande de systèmes à évènements discrets : synthèse de correcteurs pour les graphes d'événements temporisés dans les dioïdes.* PhD thesis, LISA-Université d'Angers, 1999.

[34] M. Lhommeau. *Etude de systèmes à évènements discrets dans l'algèbre (max,+). Synthèse des correcteurs robustes dans un dioide d'intervalles. Synthèse des correcteurs en présence de perturbations.* PhD thesis, Université d'Angers (LISA), 2003.

[35] J.P. Quadrat G. Cohen, D. Dubois and M. Viot. Analyse du comportement périodique des sytstèmes de production par la théorie des dios. Technical report, Rapport I.N.R. I.A N 191, 1983.

[36] J.P. Quadrat G.Cohen, D. Dubois and M. Viot. A linear-system-theoretic view of discrete event process. In *2200 IEEE Conf. on Decision and Control. San Antonio , Texas*, 1983.

[37] R. Cuninghame-Green. *Minimax algebra.* Springer-Verlag, Berlin, Germany, 1979.

[38] M. Gondran and M. Minoux. *Graphs and algorithms.* Wiley Interscience, 1984.

[39] G. Cohen, P. Moller, J.P. Quadrat, and M. Viot. Dating and counting events in discrete-event systems. In *Proceedings of the 25th Conference on Decion and Control, Athens, Greece*, 1986.

[40] G. Cohen, D. Dubois, J.P. Quadrat, and M. Viot. Algebraic tools for the performance evaluation of discrete-event systems. *Proceedings of the IEE*, 77(1) :39–57, 1989.

[41] M. Al Saba. *Contribution à la commande des systèmes flexibles de production manufacturières dans l'algèbre (max,+).* PhD thesis, Université d'Angers, 2006.

[42] G. Cohen, S. Gaubert, R. Nikoukhah, and J.P. Quadrat. Convex analysis and spectral analysis of timed event graphs. In *Proceedings of the 28th IEEE Conference on Decision and Control, Tampa, Florida*, 1989.

[43] G. Cohen, D. Dubois, J.P. Quadrat, and M. Viot. A linear-system-theoretic view of discrete-event processes and its use for performance evaluation in manufacturing. *IEEE Transactions on Automatic Control*, AC-30 :210–220, 1985.

[44] N.G. Odrey and G.R. Wilson. An on-line control structure for flexible manufacturing systems. In *Proceedings of the 14th Conference on Production Research and Technology, University of Michigan*, October, 1987.

[45] A. Saleh. *Real time control of flexible manufacturing cells.* PhD thesis, Lehigh University, Department of Industrial Engineering, 1988.

[46] J.A.C. Resing, R.E. de Vries, G. Hooghiemstra, M.S. Keane, and G.J. Olsder. Asymptotic behavior of random discrete event systems. In *Proceedings of the 28th IEEE Conference on Decision and Control, Tampa, Florida, 1532-1538*, December 1989.

[47] G.J. Olsder, J.A.C. Resing, R.E. De Veries, M.S. Keane, and G. Hooghiemstra. Discrete event systems with stochastics processing times. *IEEE Transactions on Automatic Control*, 35, N. 3 :299–302, March 1990.

[48] R. Rachamadugu and K.E. Stecke. Classification and review of fms scheduling procedures. Technical report, The University of Michigan, 1989.

[49] G. Hetreux. *Structure de décision multi-niveaux pour la planification de la production : robustesse et cohérence des décisions.* PhD thesis, Université Paul Sabatier de Toulouse, France, 2003.

[50] N.G. Odrey and G.R. Wilson. Hierarchical planning and control for flexible manufacturing shop. In *Conference Proceedings : NFS Design and Manufacturing Garantees Conference, Arizona State University, Tempe, AR*, 1990.

[51] J. Carlier and E. Pinson. An algorithm for solving the job-shop problem. *Management Science*, N 35 :164–176, 1989.

[52] R. Belemann and S. E. Dreyfus. *Applied dynamic programming.* Princeton university press, 1974.

[53] G. Bel and J.B. Cavaill*Approche simulatoire. Chapitre 6 de : Ordonnancement de la production.* Herm France, 2001.

[54] X. R Cao and Y. C Ho. Models of discrete event dynamic systems. *IEEE Control Systems Magazine*, 10(4) :69–76, 1990.

[55] P. Moller. *Théorie algébrique des systèmes à évènements discrets.* PhD thesis, Ecole nationale supérieur des Mines de Paris, 1988.

[56] G. Cohen, P. Moller, J.P. Quadrat, and M. Viot. Linear system theory for discrete event systems. In *Proceedings of the 23rd IEEE Conference on Decision and Control, Las Vegas*, December 1984.

[57] K. Inan and P. Varaiya. Finitely recursive process models for discrete event systems. *IEEE Transactions on Automatic Control*, 33, No. 7 :626–639, July 1988.

[58] B. De Schutter and T.J.J. Van Den Boom. Model predictive control for max-plus-linear discrete-event systems : Extended report & addendum. *Delft, Netherlands : Delft University of Technology*, 2000.

[59] L. Houssin. *Contribution à la commande des synthèse (max,+)-linéaires : Applications aux réseaux de transport.* PhD thesis, Université d'Angers (ISTIA), 2006.

[60] L. Houssin, S. Lahaye, and J. L Boimond. Just in time control of constrained (max,+)-linear systems. *Discrete Event Dynamic Systems*, 17(2) :159–178, 2007.

[61] S. Engell. A decentralized on-line scheduling strategy for fms. In *Sixth IFAC/IFIP/IFORS/IMACS Symposium on Information Control Problems in Manufacturing Technology, Madrid*, 1989.

[62] S. Engell, T. Kuhn, and M. Moser. On decentralized on-line scheduling of fms. In *Proceedings of the 29th Conference on Decision and Control, Honolulu, Hawaii*, 1990.

[63] D.D. Cofer and V.K. Garg. A timed model for the control of discrete event systems involving decisions in the max-plus algebra. In *Proceedings of the 31st Conference on Decision and Control, Tucson, Arizona*, 1992.

[64] D.D. Cofer and V.K. Garg. A generalized max-algebra model for performance analysis of timed and untimed discrete event systems. In *Proceedings of the American Control Conference, San Francisco, California*, 1993.

[65] I. Nasri, G. Habchi, and R. Boukezzoula. Scheduling and control modeling of HVLV systems using max-plus algebra. In *5th International Workshop on Verification and Evaluation of Computer and Communication Systems (VECoS'11), Tunis, Tunisia, 15-16 Septembre 2011. Pages 62-70*, 2011.

[66] N. Zribi, A.E. Kamel, and P. Borne. Minimizing the makespan for the mpm job-shop with availability constraints. *International Journal of Production Economics*, 112 :151–160, 2008.

[67] Mohammed Sbihi and Christophe Varnier. Single-machine scheduling with periodic and flexible periodic maintenance to minimize maximum tardiness. *Computers & Industrial Engineering*, 55(4) :830 – 840, 2008.

[68] I. Nasri, R. Boukezzoula, and G. Habchi. A mathematical model for hvlv systems scheduling and optimization with periodic preventive maintenance using (max, +) algebra. In *14th IFAC Symposium on Information Control Problems in Manufacturing (INCOM 2012), Bucharest, Romania*, 2012.

[69] I. Nasri, G. Habchi, and R. Boukezzoula. An algebraic max-plus model for hvlv systems scheduling and optimiszation with repetitive and flexible periodic preventive maintenance : just-in-time production. In *9ème Conférence Internationale de Modélisation, Optimisation et SIMulation (MOSIM 2012), Bordeaux, France*, 2012.

[70] J.L. Boimond and J.L. Ferrier. Internal model control and max-algebra : Controller design. *IEEE Transactions on Automatic Control*, 41, No 3 :457–461, 1996.

[71] A.Guezzi. *Modélisation, analyse de performance et commande des systèmes à évènements discrets*. PhD thesis, Université d'Angers, 2010.

[72] Ph. Declerck. *Systèmes à évenements discrets dans l'algèbre des dios et l'algèbre conventionnelle*. PhD thesis, université d'angers, 2011.

[73] P. Aygalinc, S. Calvez, W. Khansa, and S. Collart-Dutilleul. Using p-time petri nets for robust control of manufacturing systems. In *1 st IFAC-Workshop on Manufacturing Systems (MIM'97), Wien, austria*, 1997.

[74] Ph. Declerck and K. Didi Alaoui. External trajectories in p-time event graphs. application to control synthesis with specifications. In *44th IEEE conference on Decision and Control and European Control Conference, ECC, Seville Spain*, 2005.

[75] Ph. Declerck and M.K. Didi Alaoui. Optimal control synthesis in interval descriptor systems, application to time stream event graphs. In *IFAC congress, Praha*, 2005.

[76] B. De Schutter and T.J.J. Van Den Boom. Model predictive control for max-plus-linear systems. In *In Proceedings of American Control Conference 2000, pages 4046-4050*, 2000.

[77] A. Guezzi, Ph. Declerck, and J.-L. Boimond. From monotone inequalities to model predictive control. In *13th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), Hamburg*, 2008.

[78] D.W. Clarke, C. Mohtadiand, and P.S. Tuffs. Generalized predictive control- part 1. the basic algorithm. *Automatica*, 23(2) :137–148, 1987.

[79] D.W. Clarke, C. Mohtadiand, and P.S. Tuffs. Generalized predictive control- part 2. extensions and interpretations. *Automatica*, 23(2) :149–160, 1987.

[80] P. Boucher and D. Dumur. *La commande prédictive*. 1996.

[81] J. Richalet, G. Lavielle, and J. Mallet. *La commande prédictive*. 2004.

[82] B. De Schutter and T.J.J. Van Den Boom. On model predictive control for max-min-plus-scaling discrete event systems. *Automatica*, 37(7) :1049–1056, 2001.

[83] B. De Schutter and T.J.J. Van Den Boom. Model predictive control for max-plus-linear discrete event systems. *Automatica*, 37(7) :1049 – 1056, 2001.

[84] B. De Schutter and T.J.J. Van Den Boom. MPC for discrete-event systems with soft and hard synchronization constraints. *International Journal of Control*, 76(1) :82–94, 2003.

[85] B. De Schutter and T.J.J. Van Den Boom. MPC for continuous piecewise-affine systems. *Systems & Control Letters*, 52(3-4) :179 – 192, 2004.

[86] I. Nasri, G. Habchi, and R. Boukezzoula. Max-plus-linear model-based predictive control for constrained HVLV manufacturing systems. In *16th IEEE International Conference on Emergency Technologies and Factory Automation (ETFA'11), Toulouse, France, 5-9 Septembre 2011. Pages 1-4*, 2011.

[87] I. Nasri, R. Boukezzoula, and G. Habchi. A propos de la modélisation et le pilotage des systèmes manufacturiers de type HVLV. In *12ème Congrès Annuel de la Société Française de Recherche Opérationnelle et d'aide à la Décision (ROADEF'11), Saint-Etienne, France, 2-4 March, N383, Page II-665*, 2011.

[88] M. Pinedo. *Scheduling theory, algorithms, and systems.* 1995.

[89] Gotha. Groupe flexibilit gotha (2002), flexibilit robustesse en ordonnancement. In *Bulletin semestriel de la ROADEF vol 9*, 2002.

[90] P. Esquirol and P. Lopez. *Concepts et méthodes de base en ordonnancement de la production. Ordonnancement de la production.* 2001.

[91] D. Berkoune. *Optimisation de l'ordonnancement prenant en compte les tâches prévisionnelles.* PhD thesis, Université de Valenciennes et du Hainaut Cambrésis, 2005.

[92] J. Carlier and P. Chretienne. *Problèmes d'ordonnancement modélisation/complexité/Algorithmes.* Edition Masson, Paris, 1988.

[93] P. Esquirol and P. Lopez. *L'ordonnancement. Série : production et techniques quantitatives appliquées à la gestion.* 1999.

[94] P. Brucker. *Scheduling Algorithms.* 2003.

[95] A. Jain and S. Meeran. A state-of-the-art review of job-shop scheduling techniques. Technical report, Departement of applied physics, electronic and mechanical engineering, University of Dundee, Dundee, Scotland, 1998.

[96] S. B. JR. Akers and J. Freidman. A non numerical approach to scheduling problems. *Operations Research*, 3 :429–442, 1955.

[97] J. R. Jackson. An extension of johnson's result on job-lot scheduling. *Naval Research Logistics/Quarterly*, 3, 1956.

[98] S. M. Jhonson. Optimal two and three stage production schedules with set-up times included. *Naval Research Logistics/Quarterly*, 1 :61–68, 1954.

[99] B. Roy and B. Sussmann. Les problèmes d'ordonnancement avec contraintes disjonctives. Technical report, Notes D. S. 9 bis, SEMA, Paris, France, 1996.

[100] J. Carlier and E. Pinson. An algorithm for solving the job shop problem. *Management Science*, 35 :164–176, 1989.

[101] J. Carlier and E. Pinson. A practical use of jackson's pre-emptive schedule for solving the job-shop problem. *Annals of Operations Research*, 26 :269–287, 1990.

[102] S. S. Panwalkar and W. Iskander. A survey of scheduling rules. *Operations Research*, 25 :45–61, 1977.

[103] J. Adams, B. Egon, and Z. Daniel. The shifting bottleneck procedure for job shop scheduling. *Management Science*, 34 :391–401, 1988.

[104] E. Taillard. Parallel tabu search techniques for the job shop scheduling problem. *ORSA Journal on Computing*, 16 :108–117, 1994.

[105] T. Yamada and R. Nakano. Job shop scheduling by simulated annealing combined with deterministic local search. In *Metaheuristics International Conference, Hilton, Colorado, USA*, 1995.

[106] H. Jonsson. *Neural network approaches to job shop problems.* PhD thesis, Lund University, 1997.

[107] S. V. D. Zwaan and C. Marques. Ant colony optimisation for job shop scheduling. In *Proceedings of the Third Workshop on Genetic Algorithms and Artificial Life*, 1999.

Résumé : Les travaux présentés dans cette thèse portent sur la conception d'une méthodologie d'ordonnancement/optimisation pour les systèmes de production à grande variété de produits et faible densité de flux appelés systèmes HVLV (*High-Variety, Low-Volume*). Les caractéristiques de ces systèmes nous permettent d'appréhender la représentation des flux y circulant par un modèle discret. Le comportement discontinu des systèmes HVLV peut être caractérisé par la connaissance des dates de début et de fin des activités de production. L'algèbre (max, +) est utilisée pour représenter ce type de systèmes où les relations entre les dates de début des activités nécessitent l'utilisation des opérateurs maximum et addition. Afin d'utiliser l'algèbre (max, +) pour l'ordonnancement des systèmes HVLV, il est indispensable de résoudre un problème de conflit et d'optimisation sous contraintes dans cette algèbre. D'abord, nous avons développé dans ces travaux de recherche un modèle d'ordonnancement (max, +) pour les systèmes HVLV dans lequel des variables de décision ont été introduites afin de résoudre le problème de conflit entre les opérations exécutées sur les machines. Ensuite, nous avons amélioré le modèle proposé pour tenir compte de la maintenance préventive. Deux types de maintenance ont été considérés : Maintenance Périodique Répétitive (MPR) et Maintenance Flexible Périodique (MFP). Dans les deux cas, un problème d'ordonnancement non-linéaire sous contraintes a été résolu afin de minimiser certains critères de performance. Enfin, la méthodologie proposée a été validée par simulation, sur des systèmes HVLV complexes de type job-shop.
Mots-clefs : Systèmes HVLV, modèle d'ordonnancement (max, +), variables de décision, optimisation non-linéaire, simulation, maintenance préventive.

Abstract: This thesis deals with the development of a flow scheduling/optimization approach applied to the field of high-variety, low-volume production systems called HVLV (High-Variety, Low-Volume) systems. In this context, the flow of parts is represented by a discreet flow model. The discontinuous behavior of HVLV systems can be characterized by the knowledge of the starting and ending times of its activities. (Max, +) algebra is used to represent these kinds of systems where relationships between the starting times of the activities require both the maximum and addition operators. In order to use (max, +) algebra for HVLV systems scheduling, it is necessary to solve into this algebra an optimization problem subject to conflicts and constraints. In this research, we have first of all developed a scheduling (max, +) model for HVLV systems where decision variables are introduced to solve the conflict problem between operations carried out on the machines. Then, we have improved the proposed model to deal with preventive maintenance. Two kinds of maintenance are considered: Repetitive Periodic Maintenance (RPM) and Flexible Periodic Maintenance (FPM). In both cases, a non-linear optimization problem with constraints is solved to minimize some performance criteria. Lastly, simulation results on some complex HVLV job-shop systems are presented to illustrate the feasibility of the proposed methodology.
Keywords: HVLV systems, (max, +) scheduling model, decision variables, nonlinear optimization, simulation, preventive maintenance.