État de l'art en planification et ordonnancement

Ce chapitre propose un état de l'art sur les problèmes de dimensionnement de lots avec capacité, i.e., les problèmes de planification tactique qui s'intéressent dans une certaine mesure à la faisabilité du plan de production au niveau opérationnel. Parmi ces problèmes, nous mettons l'accent sur les problèmes intégrant des contraintes d'ordonnancement ou de capacité détaillée dans la formulation mathématique, garantissant des solutions réalisables.

2.1 Introduction

Dans le chapitre précédent, nous avons présenté une vision globale du fonctionnement de la chaîne logistique, en expliquant le rôle des différents acteurs, les enjeux et l'importance d'avoir une stratégie de management intégrant les différentes étapes du cycle de planification (approvisionnement, production, distribution et ventes). Nous avons présenté plus particulièrement les sous-étapes de planification (au niveau tactique) et d'ordonnancement de la production (au niveau opérationnel), dont l'intégration des décisions constitue l'objet de cette thèse.

Nous nous intéressons dans ce chapitre aux problèmes de planification de la production discrets, plus précisément aux problèmes de dimensionnement de lots avec contraintes de capacité, où la prise en compte d'informations sur l'ordonnancement est importante, afin de garantir des solutions réalisables ou d'améliorer la qualité des décisions.

Nous commençons ce chapitre avec des généralités sur les problèmes d'optimisation, dans la Section 2.2, nécessaires à la compréhension des détails sur les problèmes de dimensionnement de lots et d'ordonnancement qui sont présentés par la suite. Dans la Section 2.3, nous décrivons la classification habituelle des problèmes de dimensionnement de lots, et nous détaillons les caractéristiques principales des problèmes avec prise en compte de la capacité de production, qui sont étudiés dans la littérature, ainsi que les travaux les plus importants sur chacun de ces types de problèmes. Ensuite, dans la Section 2.4, nous présentons brièvement les caractéristiques des problèmes d'ordonnancement les plus étudiés dans la littérature. Dans la Section 2.5, nous discutons de l'importance d'intégrer les décisions de planification et d'ordonnancement au niveau tactique, et nous présentons les travaux qui intègrent de manière cohérente ces décisions, selon une classification basée sur la nature de l'approche de résolution. Enfin, nous terminons ce chapitre par une conclusion sur les différentes thématiques traitées.

2.2 Généralités sur les problèmes d'optimisation

Nous avons brièvement discuté, dans le chapitre précédent, la difficulté associée à la résolution de plusieurs problèmes, avec objectifs différents, dans la planification de la chaîne logistique, et particulièrement lorsque l'on veut suivre une approche d'optimisation globale, comme établie par la matrice de planification des logiciels APS. Les problèmes associés aux étapes de planification de la chaîne logistique, dans tous les niveaux de décisions, peuvent être modélisés comme des problèmes d'optimisation, ayant une fonction objectif (par exemple, la minimisation du coût, la minimisation du retard, la minimisation du nombre de véhicules, etc.) et des contraintes (par exemple, la satisfaction de la demande, une capacité de production limitée, des

lots de production avec une taille maximale, des véhicules limités, des fenêtres de temps pour la production ou pour la distribution, etc.). En pratique, avec toutes les données et les contraintes présentes dans les différents types de chaîne logistique, chacun des problèmes modélisant les étapes de planification, devient très difficile à résoudre, cette difficulté étant mesurable par un indicateur connu comme la *complexité*, que nous discutons dans ce qui suit. L'intégration ou la considération des contraintes de plusieurs étapes de planification dans un seul problème augmente encore plus la difficulté de résolution, raison pour laquelle, il est souvent impossible de déterminer la solution optimale d'un problème, et même d'obtenir une solution réalisable. C'est pourquoi les problèmes d'optimisation liés à la chaîne logistique nécessitent de combiner des techniques de recherche opérationnelle et la conception d'algorithmes sophistiqués, capables de résoudre des problèmes complexes dans des temps de calcul raisonnables et avec une utilisation limitée de ressources (processeur et mémoire physique des ordinateurs).

2.2.1 Complexité

La complexité est un indicateur de l'effort de calcul, par rapport au temps d'exécution, nécessaire pour la résolution d'un problème d'optimisation. Une autre mesure de l'effort de calcul est l'utilisation ou la consommation de mémoire physique (RAM dans les ordinateurs) nécessaire pour garder les données des variables, issues de l'exploration de l'espace de solution (ou espace de recherche), dans une méthode d'optimisation. En théorie de la complexité [62], on peut distinguer la complexité de l'algorithme et la complexité du problème.

La complexité du temps de calcul d'un algorithme est donnée par le nombre d'étapes nécessaires pour résoudre le problème. Étant donné que ce nombre peut varier en fonction de l'instance étudiée, la complexité de l'algorithme est définie par rapport au pire cas. La notation O est utilisée pour représenter cette complexité. Ainsi, un algorithme est applicable en temps polynomial si sa complexité est $O\left(p\left(n\right)\right)$, ou $p\left(n\right)$ est une fonction polynomiale de la taille de l'instance n (par exemple, le nombre de produits, le nombre de machines ou le nombre de périodes de planification). Donc, si $p\left(n\right) = a_k n^k + \cdots + a_j n^j + \cdots + a_1 n + a_0$, avec $a_k > 0$ et $a_j \geq 0 \ \forall \ 1 \leq j \leq k-1$, l'algorithme a une complexité polynomiale de $O\left(n^k\right)$ [212]. Des exemples de temps de calcul polynomiaux sont les complexités $O\left(n\right)$, $O\left(n \log n\right)$, $O\left(n^2\right)$, $O\left(n^3\right)$, etc. En revanche, un algorithme est exécutable en temps exponentiel si sa complexité est $O\left(c^n\right)$, où c est une constante réelle supérieure à 1.

La complexité d'un problème est équivalente à la complexité du meilleur algorithme capable de le résoudre. S'il existe un algorithme pouvant résoudre le problème en temps polynomial, le problème est dit *facile*. Sinon, le problème est considéré *difficile*. Un problème d'optimisation

peut être réduit à un problème de décision, dont la solution est de type binaire, c'est-à-dire ayant pour réponse *oui* ou *non*. Selon la complexité, les problèmes peuvent être organisés en plusieurs classes [62], dont les plus grandes et importantes sont $\mathcal{P}$ et $\mathcal{NP}$. Une classe de complexité représente l'ensemble des problèmes qui peuvent être résolus avec une quantité déterminée de ressources. Ainsi, la classe $\mathcal{P}$ fait référence à l'ensemble de tous les problèmes de décision qui peuvent être résolus en temps polynomial, au moyen d'un algorithme déterministe. Un algorithme est déterministe si le nombre d'étapes nécessaires pour résoudre le problème peut être calculé à l'avance. D'autre part, la classe $\mathcal{NP}$ est associée à l'ensemble des problèmes de décision qui peuvent être résolus en temps polynomial par un algorithme non déterministe. La caractéristique d'un algorithme non déterministe est que l'on ne peut pas prédire avec certitude le nombre d'étapes nécessaires pour résoudre le problème.

Une autre classe importante, contenue dans la classe $\mathcal{NP}$ est celle des problèmes $\mathcal{NP}$-complets. Un problème de décision G est $\mathcal{NP}$-complet si tous les problèmes de la classe $\mathcal{NP}$ peuvent être ramenés (réduits) en temps polynomial à G. Cela implique que l'on peut construire une instance de G, à partir de n'importe quelle instance d'un problème $H \in \mathcal{NP}$, suivant une fonction polynomiale de la taille de l'instance de H. Le problème d'optimisation correspondant à un problème de décision $\mathcal{NP}$-complet est dit $\mathcal{NP}$-difficile, pour lequel la résolution se fait en temps exponentiel, c'est-à-dire que le temps d'exécution augmente de façon exponentielle en fonction de la taille de l'instance. La relation entre les classes mentionnées est illustrée dans la Figure 2.1.

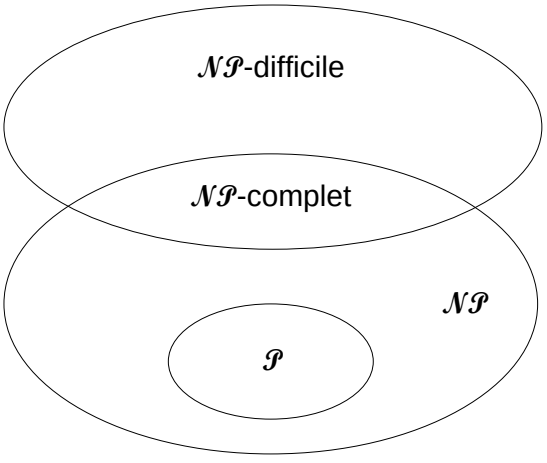

FIGURE 2.1 – Relation entre les principales classes de complexité

Nous pouvons encore distinguer entre problèmes $\mathcal{NP}$-difficiles au *sens faible* et problèmes $\mathcal{NP}$-difficiles au *sens fort*. Les premiers sont des problèmes pour lesquels il existe un algorithme

capable de les résoudre en temps polynomial, pas en fonction de la taille de l'instance, mais de la longueur des paramètres (par exemple, les demandes et les durées opératoires). Ce type d'algorithme est appelé *pseudo-polynomial*. S'il n'existe pas un tel algorithme, le problème est $\mathcal{NP}$-difficile au sens fort.

Un très grand nombre de problèmes d'optimisation associés aux étapes de planification de la chaîne logistique sont $\mathcal{NP}$-difficiles. C'est le cas, par exemple, des problèmes standard de dimensionnement de lots avec contraintes de capacité, d'ordonnancement avec plus de deux machines, de tournées de véhicules et de localisation d'usines. Dans certains cas, la simple recherche d'une solution réalisable est aussi $\mathcal{NP}$-difficile.

2.2.2 Modélisation

Nous nous intéressons, dans ce travail, à la modélisation par programmation mathématique. D'autres types de modélisations, que nous ne décrivons pas dans ce travail, sont les modèles d'optimisation combinatoire, les modèles de satisfaction de contraintes et les modèles non analytiques.

En programmation mathématique, les modèles peuvent être classés en : problèmes en nombres réels (continuous programs), problèmes en nombres entiers (IP ou integer programs) et problèmes en nombres entiers mixtes (MIP ou mixed integer programs). Le nom de chaque type de modèle fait référence à la nature des variables de décision utilisées dans la formulation. Ainsi par exemple, les modèles MIP comportent des variables de décision réelles (ou continues) et entières (ou discrètes). Ce type de modélisation peut être vue comme une sous-catégorie de la modélisation IP, car les méthodes de résolution pour des problèmes à nombres réels ne donnent pas des solutions réalisables pour ces modèles, contrairement aux méthodes dédiées pour des problèmes à nombres entiers. Un autre sous-ensemble des modèles IP est celui des modèles 0-1, où les variables de décision prennent uniquement des valeurs binaires (0 ou 1). Les modèles avec uniquement des variables de décision réelles sont plus faciles à résoudre, avec une complexité polynomiale quand la fonction objectif et les contraintes sont linéaires.

La plupart des problèmes de dimensionnement de lots et d'ordonnancement sont modélisés par des programmes à variables mixtes, les variables entières étant souvent binaires, comme la variable de setup Y_{il} qui vaut 1 si la production du produit i est lancée à la période l, ou 0 sinon. En général, l'inclusion de variables binaires dans la formulation mathématique rend plus difficile la résolution du problème. C'est pourquoi, des méthodes de relaxation et de décomposition sont utilisées pour ne pas considérer ou pour fixer ces variables.

Les modèles mathématiques peuvent être linéaires ou non linéaires. Dans le premier cas, la fonction objectif et toutes les contraintes sont linéaires. Le terme de programmation linéaire (LP pour Linear Programming) est associé aux modèles linéaires en nombres réels. Dans les autres cas, on parle de programmation linéaire en nombres entiers (ILP pour Integer Linear Programming) et programmation linéaire en nombres entiers mixtes (MILP pour Mixed Integer Linear Programming). La plupart des problèmes de dimensionnement de lots et d'ordonnancement sont formulés comme des modèles linéaires, et plus précisément comme des MILP. Pochet et Wolsey [185] présentent de manière détaillée des formulations MIP, qui sont utilisées pour modéliser un grand nombre de problèmes de dimensionnement de lots étudiés dans la littérature.

Une manière particulière de modéliser les problèmes d'ordonnancement est au moyen d'un **graphe disjonctif**, modèle proposé par Roy et Sussmann [194] en 1964, plus particulièrement pour le problème dans un atelier de type job-shop. Un graphe disjonctif $G = (V, C, D)$ est composé d'un ensemble V de sommets représentant les opérations ou tâches de production, d'un ensemble C d'arcs conjonctifs représentant les contraintes de précédence entre les opérations d'une même gamme de fabrication, et d'un ensemble D d'arcs disjonctifs représentant les contraintes de séquencement entre les opérations traitées sur la même ressource. L'objectif est de trouver la meilleure orientation possible pour chaque arc disjonctif, sans créer de cycles, de façon à optimiser un critère pouvant être : la minimisation du temps de cycle, la minimisation du nombre de jobs en retard, etc., comme expliqué dans la Section 1.5 du Chapitre 1. Un exemple de graphe disjonctif est illustré dans la Figure 3.1 dans la Section 3.2 du Chapitre 3.

2.2.3 Méthodes de résolution

La résolution des problèmes d'optimisation est basée sur des techniques de recherche opérationnelle, dont nous distinguons les méthodes exactes et les méthodes approchées. Les méthodes exactes ont pour but de déterminer la solution optimale du problème (optimum global), tandis que les méthodes approchées s'intéressent à l'obtention de la meilleure solution possible, mais dont l'optimalité n'est pas garantie. Les méthodes exactes demandent souvent un effort de calcul très important. C'est pourquoi, les méthodes approchées, dont l'objectif est de considérablement réduire l'effort de calcul en diminuant l'espace des solutions à explorer, sont souvent appliquées pour résoudre des problèmes $\mathcal{NP}$-difficiles, comme la plupart des problèmes de dimensionnement de lots avec contraintes de capacité.

2.2.3.1 Méthodes exactes

Pour des problèmes linéaires en nombres réels, la méthode la plus utilisée est l'algorithme du **simplexe** [157], proposé par George Dantzig, en 1947. Cette méthode ne donne pas des solutions réalisables pour les problèmes à nombres entiers ou à nombres entiers mixtes, mais elle peut être incorporée dans des méthodes résolvant des problèmes où les contraintes imposant des valeurs entières sont relâchées. Les problèmes de dimensionnement de lots et d'ordonnancement, étant modélisés le plus souvent par des programmes MIP, nous nous intéressons aux méthodes de résolution pour ce type de modèles. Parmi les méthodes exactes les plus utilisées, nous trouvons : la décomposition de Dantzig-Wolfe, la décomposition de Benders, le *branch and bound*, le *branch and cut*, le *branch and price* et la programmation dynamique. D'autres méthodes, que nous ne détaillons pas ici, sont la programmation par contraintes [192] et les méthodes A* [108] et IDA* [139].

La **décomposition de Dantzig-Wolfe** a été proposée par Dantzig et Wolfe [45] en 1960. Cette méthode peut être appliquée aux problèmes avec contraintes compliquées, ayant une structure de bloc angulaire, c'est-à-dire, ceux où l'ensemble de contraintes peut être décomposé en deux sous-ensembles. Un ensemble correspond aux contraintes qui ne lient pas des variables différentes, et l'autre ensemble est associé aux contraintes couplantes (liant plusieurs variables). Le problème original est décomposé en un problème maître et plusieurs sous-problèmes de taille réduite. Cette méthode est une version améliorée de l'algorithme du *simplexe*, utilisant la génération de colonnes [54]. Cette dernière approche est utilisée pour résoudre des problèmes de grande taille, où le problème est initialisé avec un sous-ensemble de colonnes (variables) de taille réduite, et, à chaque itération, une colonne pouvant potentiellement améliorer la solution courante est ajoutée. Pour plus de détails sur la décomposition de Dantzig-Wolfe, le lecteur peut consulter [41]. Les références [52] et [76] sont des exemples d'utilisation de cette méthode dans des problèmes de dimensionnement de lots et d'ordonnancement, respectivement.

La **décomposition de Benders**, proposée par Benders [28] en 1962, est une méthode de décomposition pour des problèmes de grande taille, ayant une structure de bloc, avec des variables compliquées. Contrairement à la décomposition de Dantzig-Wolfe, qui applique la génération de colonnes, la décomposition de Benders ajoute des contraintes progressivement. Pour simplifier la résolution du problèmes, les variables compliquées sont fixées de façon dynamique dans une méthode itérative. Dans les problèmes MILP, la décomposition de Benders est utilisée pour fixer les variables binaires à des valeurs spécifiques réalisables et résoudre le problème LP résultant de façon optimale. Pour plus de détails, le lecteur peut consulter [41]. Parmi les travaux résolvant un problème de dimensionnement de lots au moyen de la décomposition de

Benders, nous pouvons citer [1]. Pour les problèmes d'ordonnancement, le lecteur peut consulter [153].

Le **branch and bound** (B&B), proposé par Land et Doig [142] en 1960, est une méthode de recherche arborescente appliquée en deux étapes de façon itérative. La première étape est la *séparation*, qui consiste à sélectionner les branches de l'espace de solution à explorer, en décomposant un sommet de l'espace des solutions en plusieurs sous-ensembles disjoints de plus petite taille. La deuxième étape est l'*évaluation* et consiste à énumérer systématiquement les solutions candidates, dont les mauvaises sont écartées, sans résoudre de façon détaillée chaque sous-problème. En fait, le choix se base sur les valeurs de la borne inférieure (dans le cas d'un problème de minimisation) ou de la borne supérieure (dans le cas d'un problème de maximisation), en résolvant le sous-problème relâché. À chaque itération, des nouvelles branches sont explorées. Le problème relâché peut par exemple être résolu par une relaxation linéaire, où les contraintes imposant des valeurs binaires à certaines variables sont enlevées, ou par une relaxation Lagrangienne, comme expliqué plus tard dans cette Section. Pour plus de détails sur le B&B, le lecteur peut consulter [35]. Cette technique a été amplement utilisée pour la résolution de problèmes de dimensionnement de lots et d'ordonnancement. À titre d'exemple, nous citons [75] et [53].

Le **branch and cut** (B&C) est une généralisation du B&B. Il suit aussi les étapes de séparation et évaluation, mais avec une variante. La méthode des plans sécants (*cutting plane algorithm*) [236] est utilisée pour définir une nouvelle contrainte, permettant de rendre entières les valeurs de variables relâchées, dont la solution initiale (au moyen de l'algorithme du *simplexe*) est réelle. La génération de plusieurs contraintes peut être nécessaire afin d'obtenir une solution réalisable. Une fois qu'une telle solution est trouvée, les étapes de séparation et évaluation sont appliquées. Le problème est séparé en deux sous-problèmes, à partir de la nouvelle contrainte, et chaque sous-problème est résolu à nouveau par l'algorithme du *simplexe*. La procédure continue de façon itérative jusqu'à converger vers une solution optimale. Si les contraintes sont ajoutées uniquement dans le modèle initial, la méthode prend le nom de *cut and branch* (C&B). Un problème d'ordonnancement résolu par un B&C est présenté dans [190]. La méthode est appliquée pour résoudre un problème de dimensionnement de lots dans [96].

Le **branch and price** (B&P) est aussi une généralisation du B&B. Contrairement au B&C, le *branch and price* n'ajoute pas des contraintes, mais des variables, au moyen de la génération de colonnes. Au début de l'algorithme, un ensemble de variables (colonnes) est enlevé de la modélisation initiale. Des nouveaux problèmes sont résolus au fur et à mesure, en rajoutant des colonnes à chaque itération. Une description détaillée de cette méthode est présentée

dans [26]. Des exemples d'utilisation de cette technique dans des problèmes de planification et d'ordonnancement de la production peuvent être consultés dans [52] et [179], respectivement.

La **programmation dynamique**, proposée par Bellman [27], en 1957, est une méthode de « force brute », c'est-à-dire qu'elle explore toutes les alternatives possibles, permettant de trouver la solution optimale par recherche exhaustive. Le problème original est décomposé en plusieurs sous-problèmes plus faciles à résoudre. Le principe de cette méthode est que la solution de chaque sous-problème est gardée en mémoire, de telle sorte que si un sous-problème est composé ou requiert le calcul de la solution d'un sous-problème préalablement résolu, la résolution n'est pas refaite, permettant d'économiser en temps de calcul. Une description plus détaillée de cette méthode peut être consultée dans [149]. À titre d'exemple, la programmation dynamique est utilisée dans [199] pour résoudre un problème de dimensionnement de lots et dans [125] pour résoudre un problème d'ordonnancement.

2.2.3.2 Méthodes approchées

Il existe cinq grands groupes de méthodes approchées : les heuristiques basées sur la programmation (modèle) mathématique, les heuristiques Lagrangiennes, les heuristiques de décomposition et d'agrégation, les métaheuristiques et les heuristiques gloutonnes spécifiques au problème. Nous nous basons, dans ce qui suit, sur la classification réalisée par Buschkühl *et al.* [36].

Les **heuristiques basées sur la programmation mathématique** sont conçues à partir de méthodes exactes. Une manière simple de concevoir une telle heuristique consiste à limiter le temps d'exécution ou le nombre d'itérations d'une méthode exacte. C'est le cas de l'heuristique *truncated MIP*, qui limite le temps du B&B ou du B&C. Une autre stratégie consiste à arrêter la procédure lorsque l'écart entre les bornes inférieure et supérieure est inférieure ou égal à une certaine valeur définie avant de démarrer la procédure. Donc, si ces heuristiques suivent le principe d'une méthode exacte, elle ne garantissent pas l'obtention d'une solution optimale. Parmi ces heuristiques, nous trouvons des procédures de résolution basées sur le B&B, le B&C, le B&P, la décomposition de Dantzig-Wolfe et la décomposition de Benders. D'autres heuristiques sont les suivantes :

- *LP and fix*. C'est une méthode itérative où, à chaque itération, les variables binaires sont fixées, et le problème linéaire correspondant est résolu. Cette heuristique est appelée *cut and fix* quand le modèle mathématique est amélioré (réduit) par des coupes ou par une reformulation. Une application de cette méthode sur un problème de dimensionnement

de lots peut être consultée dans [202].

- *Fix and relax* (ou *relax and fix*). Cette méthode décompose le problème en plusieurs sous-problèmes, chacun avec un ensemble de variables binaires fixées, organisées en trois groupes. Le premier groupe est résolu de façon optimale, le deuxième est relâché et, dans le troisième groupe, les variables binaires sont fixées aux valeurs d'une itération précédente. Un travail utilisant cette heuristique pour résoudre un problème de dimensionnement de lots est présenté dans [24].

- *Fix and optimize* (ou *exchange* dans [185]). C'est une variante de la méthode *fix and relax*, avec la différence que le groupe de variables relâchées n'est pas considéré, permettant donc de déterminer une solution réalisable à chaque itération. Cette méthode est utilisée pour résoudre un problème de dimensionnement de lots dans [143].

- Heuristiques d'arrondi. Le but est de réduire la difficulté de résolution du modèle, notamment associée aux variables binaires. Ainsi, le problème est relâché par une relaxation linéaire, ou les variables binaires prennent des valeurs continues entre 0 et 1. Ces valeurs sont ensuite arrondies par rapport à un seuil. Ce type d'heuristiques est utilisé pour résoudre un problème de dimensionnement de lots dans [154].

Ces heuristiques ont été largement étudiées pour la résolution de problèmes de dimensionnement de lots avec variables de setup. D'autres stratégies de résolution basées sur la programmation mathématique sont les reformulations et les inégalités valides. Les reformulations consistent à modifier le modèle mathématique, en redéfinissant certaines variables et contraintes, avec le but de rendre le modèle plus facile à résoudre ou de générer de meilleures bornes. Deux reformulations typiques en dimensionnement de lots, spécifiquement pour le problème *Capacitated Lot Sizing Problem*, que nous décrivons plus tard dans ce chapitre, sont : le modèle du plus court chemin [216] et le modèle de localisation de sites de production [207]. Les inégalités valides sont des contraintes qui sont ajoutées pour réduire la taille de l'espace des solutions, permettant d'obtenir des meilleures bornes et de converger plus rapidement vers la solution finale. Il existe trois types d'inégalités valides, celles utilisées pour générer dynamiquement des coupes éliminant les solutions non-entières, celles du B&C et celles du C&B.

Pour plus de détails sur les méthodes basées sur des programmes mathématiques en nombres entiers mixtes (MIP), le lecteur peut consulter [185].

Les **heuristiques Lagrangiennes** sont des méthodes itératives utilisant le principe de la relaxation Lagrangienne, qui décompose, en relâchant certaines contraintes, le problème original en plusieurs sous-problèmes plus faciles à résoudre. Dans cette catégorie, nous trouvons les méthodes suivantes :

- Relaxation Lagrangienne. Les contraintes compliquées sont relâchées et, à chaque itération, les contraintes non respectées sont pénalisées dans la fonction objectif, au moyen de multiplicateurs qui peuvent être mis à jour par différentes méthodes, dont la plus utilisée est celle des sous-gradients [69]. La solution correspond à une borne inférieure, qui évolue itérativement jusqu'à converger vers le dual Lagrangien. L'idée de cette heuristique est donc de proposer une solution qui nécessite peu de modifications pour être optimale, sans avoir à résoudre le problème original, mais des sous-problèmes plus faciles. Des modifications systématiques de la solution peuvent être réalisées tout au long de la procédure, permettant de proposer des solutions réalisables, i.e. des bornes supérieures, qui peuvent être utilisées pour guider l'évolution des multiplicateurs. La relaxation Lagrangienne a été largement utilisée pour la résolution de problèmes de planification (voir par exemple [221]) et d'ordonnancement (voir par exemple [37]). Pour plus de détails sur cette méthode, le lecteur peut consulter [146].

- Décomposition Lagrangienne (ou relaxation Lagrangienne augmentée). Elle diffère de la relaxation Lagrangienne par le fait qu'aucune contrainte du problème original n'est relâchée, mais le problème est décomposé en plusieurs sous-problèmes par duplication de variables, chaque sous-problème incluant un sous-ensemble de contraintes. Pour garantir la faisabilité des solutions, les variables dupliquées sont égalisées aux variables originales, au moyen de contraintes couplantes, qui sont relâchées. Une application de cette technique sur un problème d'ordonnancement peut être consultée dans [114]. Pour ce qui est des problèmes de dimensionnement de lots, le lecteur peut consulter [218].

Les méthodes de **décomposition** et d'**agrégation** transforment le problème original en plusieurs sous-problèmes de taille réduite et plus faciles à résoudre. D'une part, les approches de décomposition séparent le problème en plusieurs sous-problèmes détaillés. D'autre part, les approches d'agrégation réduisent le niveau de détail du modèle, en rassemblant certaines données (par exemple, la demande ou la capacité). Dans les problèmes de dimensionnement de lots, la décomposition peut être faite à partir des produits [216], du temps (périodes de planification) [2] et des ressources, tandis que l'agrégation se fait seulement à partir des produits [177] et des ressources [32]. La construction d'une solution réalisable se fait au moyen de procédures itératives, permettant de modifier et de faire évoluer les valeurs des variables de décision.

Les **métaheuristiques** sont des procédures utilisées pour guider l'exploration de l'espace des solutions d'un problème complexe, de façon à trouver une solution réalisable, qui offre un bon compromis entre effort de calcul et qualité de la solution. La performance d'une métaheuristique dépend de la combinaison de deux critères opposés : la diversification et l'intensification. Le premier fait référence à l'exploration de différentes branches de l'espace des solutions, et le deuxième correspond à l'exploitation des régions prometteuses. L'intensification est une étape de recherche locale, dans laquelle on cherche à améliorer la qualité de la solution. La diversification permet de sortir de zones très explorées, pour en visiter d'autres, en évitant de rester sur la région d'un optimum local. Les métaheuristiques les plus utilisées dans les problèmes de planification et d'ordonnancement de la production sont les suivantes :

- Recherche locale. L'exploration de l'espace de solution est focalisé sur une région. Des changements systématiques sur certaines variables du problème permettent d'obtenir une nouvelle solution à chaque itération, à partir d'un voisinage de solutions candidates. Ainsi, la solution déterminée à la solution k fait partie des solutions candidates à l'itération $k-1$. La méthode standard s'arrête lorsqu'il est n'est plus possible d'améliorer la solution après un certains nombre d'itérations. La solution correspond à un optimum local. Des exemples d'utilisation de la recherche locale pour des problèmes de planification et d'ordonnancement peuvent être consultés dans [101] et [224], respectivement.

- Recuit simulé. C'est une méthode recherche globale, proposée par Kirkpatrick *et al.* [137], en 1983, et par Černý [228], en 1985, basée sur le processus du recuit en métallurgie, qui consiste à augmenter la température d'un métal (typiquement l'acier), jusqu'à ce que la matière atteigne un état spécifique (avant la fusion), et ensuite à le refroidir lentement à température contrôlée, pour que le matériau acquière des propriétés mécaniques spécifiques. L'algorithme simule le mouvement des atomes dans le matériau en fonction de la température, chaque mouvement permettant de définir une nouvelle solution. Le fait d'augmenter la température rend les atomes plus dynamiques (haut niveau d'énergie), et le refroidissement lent les rend statiques (bas niveau d'énergie). Pour chaque solution, il y a un voisinage de solutions candidates, et le but est de minimiser le niveau d'énergie. Néanmoins, pour éviter de rester sur la région d'un optimum local, alterner le refroidissement avec une augmentation de la température permet de diversifier l'exploration. Un travail utilisant le recuit simulé pour résoudre un problème de dimensionnement de lots est présenté dans [176]. Quant aux problèmes d'ordonnancement, une étude appliquant cette technique est présentée dans [226]. Pour plus de détails sur cette méthode, le lecteur

peut consulter [59].

– Recherche taboue. C'est une méthode de recherche globale, proposée par Glover [81], en 1985. Elle combine le principe de la recherche locale (intensification), à travers l'exploration d'un voisinage défini par la solution courante, et la diversification, à travers une structure de *mémoire*, gérée par une *liste taboue*, qui interdit l'accès à certains voisins préalablement explorés, afin d'explorer des nouvelles branches de l'espace des solutions. La liste taboue peut être vue, en effet, comme un vecteur gardant par ordre d'ancienneté les derniers mouvements effectués. Ils existe trois catégories de mémoire : à court terme, à moyen terme et à long terme. La mémoire à court terme est contrôlée par la taille de la liste taboue. Ainsi, une fois qu'un critère d'expiration (souvent le remplissage de la liste taboue) est atteint, l'élément le plus ancien de la liste est libéré, pouvant être réincorporé à des futurs voisinages. La mémoire à moyen terme permet de diriger l'intensification vers des régions prometteuses de l'espace de recherche, et la mémoire à long terme permet de conduire la recherche vers de nouvelles régions, lorsque l'exploration est bloquée dans une région sous-optimale, en tenant compte du critère d'aspiration, qui permet de faire sortir de la liste taboue des mouvements jugés intéressants. Des critères d'aspiration peuvent être : la meilleure solution sur les k dernières itérations, le meilleur voisin, une solution différente des solutions existantes, une solution similaire aux solutions existantes et l'influence sur l'évolution de la solution. Grâce à la mémoire, et contrairement à la recherche locale, les solutions trouvées par la recherche taboue à certaines itérations peuvent être détériorées par rapport à la solution précédente, évitant la convergence vers un optimum local. Le degré de diversification dépend, dans une grande mesure, de la taille de la liste taboue, qui sert à éviter des explorations cycliques. Des exemples de travaux utilisant la recherche taboue pour résoudre des problèmes de planification et d'ordonnancement sont [93] et [232].

– Recherche avec voisinage variable (VNS ou Variable Neighborhood Search) [104]. C'est une méthode de recherche globale, proposée par Mladenović [166], en 1997. Elle se déroule en trois étapes principales : agitation, recherche locale et mouvement. Durant la première étape, une solution du voisinage de la solution courante est sélectionnée au hasard. S'il n'y a pas d'amélioration, la taille du voisinage est augmentée. À partir de la solution voisine, une recherche locale est réalisée. Si celle-ci conduit vers une meilleure solution, le mouvement est effectué, et on construit un nouveau voisinage pour répéter la procédure. Sinon, on revient directement à la première étape. La procédure est appliquée de façon

itérative. Des approches de résolution basée sur le VNS sont présentées dans [20] pour la résolution d'un problème de dimensionnement de lots, et dans [191] pour résoudre un problème d'ordonnancement.

– Algorithmes génétiques. Ils se basent sur le principe biologique de la sélection naturelle. La procédure démarre avec une population d'individus, chacun avec un code génétique, qui représente les chromosomes. À chaque itération, une nouvelle génération d'individus est obtenue à partir de trois facteurs stochastiques : la sélection, le croisement et la mutation. Le meilleur chromosome généré est décodé, représentant la meilleure solution. Contrairement aux autres métaheuristiques décrites, les algorithmes génétiques ne comportent pas une étape de recherche locale. Une vue d'ensemble sur des algorithmes génétiques utilisés pour la résolution de problèmes de planification est présentée dans [98]. Un exemple d'application d'un algorithme génétique sur un problème d'ordonnancement est présenté dans [180].

D'autres métaheuristiques moins utilisées en planification et ordonnancement sont : l'optimisation par colonies de fourmis [58], qui se base sur le comportement de fourmis durant la recherche d'un chemin entre leur colonie et une source de nourriture, les algorithmes mémétiques [172], qui combinent des heuristiques de recherche locale avec les facteurs de croisement des algorithmes génétiques, la méthode *threshold accepting* [63] qui est une variante du recuit simulé, la recherche adaptative gloutonne (GRASP) [66], qui comporte une étape de construction gloutonne et une étape de recherche locale. Pour des plus amples détails sur les métaheuristiques présentées ici et d'autres, le lecteur peut consulter [212].

Les **heuristiques gloutonnes spécifiques au problème** sont classées en deux catégories : les heuristiques constructives et les heuristiques d'amélioration. Elles ont comme point commun qu'à chaque étape, le meilleur élément (parmi un ensemble d'éléments pouvant constituer une solution) est sélectionné. Le critère de sélection est la valeur de la fonction objectif. Dans les heuristiques constructives [99], la solution initiale est vide et, à chaque étape, un nouvel élément est ajouté à la solution. C'est le cas, par exemple, de *l'algorithme de liste* en ordonnancement, qui suit des règles de priorité (plus longue durée de fabrication ou LPT, plus courte durée de fabrication ou SPT, plus proche délai de livraison ou EDD, etc.) [109] pour construire une solution. En ce qui concerne les heuristiques d'amélioration [74], les solutions sont déterminées à partir de solutions initiales, réalisables ou non réalisables, en réalisant plusieurs mouvements.

2.3 Problèmes de dimensionnement de lots

Comme introduit dans le chapitre précédent, la production par lots est une stratégie pour planifier la production dans les systèmes à flux discrets. Le dimensionnement de lots est normalement réalisé au niveau tactique sur un horizon de planification de plusieurs périodes (en général jours ou semaines), et consiste à décider des quantités de production par produit et par période, afin de satisfaire la demande au moindre coût possible. Les quantités de produit sont appelées « lots » et sont calculées avec le but de regrouper des demandes de différentes périodes pour produire de manière anticipée, si cela permet de réduire les coûts de setup (coûts de lancement de la production) ou de production. Cependant, le fait de produire en avance génère des inventaires et, si les coûts de stockage sont élevés, il peut être plus intéressant de répartir la production sur plusieurs périodes (ce qui induit des setups) et de stocker moins. Le problème de dimensionnement de lots est donc souvent équivalent à trouver un compromis entre les coûts de setup et les coûts de stockage. D'autre part, si les ruptures de stocks sont autorisées, les coûts associés sont à considérer ; et si les ventes perdues sont autorisées, les coûts associés sont aussi à intégrer dans le processus de décision.

Les premières politiques utilisées pour dimensionner les lots consistaient, entre autres, à fixer une taille de lot constante arbitrairement, suivre la demande période par période (sans stockage de produit fini) ou fixer une quantité périodique (dans des scénarios à demande dynamique). Le premier effort pour optimiser les tailles de lots a été proposé par Harris [107] en 1913, avec la création du modèle EOQ (Economic Order Quantity ou quantité économique de commande), qui détermine une taille de lot fixe (constante) optimale en minimisant les coûts de fabrication, de stockage et de commande, dans un système à demande constante, mono-produit et sans contraintes de capacité.

Le premier algorithme permettant d'optimiser le problème de dimensionnement de lots avec demande dynamique a été proposé par Wagner et Whitin [231]. Cet algorithme (dans la suite de ce manuscrit, méthode WW) en temps quadratique ($O(T^2)$, où T est le nombre de périodes) permet de calculer des tailles de lots variables par période pour résoudre le problème mono-produit sans contraintes de capacité, et est à la base de beaucoup de travaux en dimensionnement de lots. Néanmoins, à l'époque de son développement, la puissance de calcul des ordinateurs ne permettait pas de déployer cette méthode. Plusieurs heuristiques ont été proposées pour ce problème, dont celle de Silver et Meal [206]. Plus tard, Wagelmans *et al.* [230], Aggarwal et Park [9], Federgruen et Tzur [65] et Van Hoesel *et al.* [225] ont proposé des versions améliorées de l'algorithme de programmation dynamique WW, de complexité $O(T \log T)$.

L'inconvénient de ces méthodes est que les contraintes de capacité ne sont pas considérées, ce qui implique que les solutions ne sont pas, dans beaucoup de cas, réalisables au niveau opérationnel. Cependant, l'étude des problèmes sans contraintes de capacité a jusqu'à présent suscité beaucoup d'intérêt dans la littérature (voir par exemple [5], [242] et [240]), car ils sont plus faciles à résoudre et parce que des problèmes plus complexes peuvent être décomposés et ramenés à l'étude de problèmes sans contraintes de capacité. Les problèmes avec contraintes de capacité ont été étudiés depuis les années 70, avec le but de réduire l'écart de faisabilité entre les décisions de planification prises à moyen terme et à court terme. Cependant, la grande majorité des modèles développés ne garantissent des solutions réalisables que pour des scénarios très précis, avec des ateliers avec peu de ressources. De plus, la capacité n'est en général pas modélisée de manière exacte, ce qui fait que les solutions sont sous-optimales. Un état de l'art sur les problèmes de dimensionnement de lots avec contraintes de capacité peut être consulté dans [61].

Un des facteurs les plus importants à considérer dans un problème de dimensionnement de lots avec contraintes de capacité est la structure des setups. Comme expliqué par Karimi *et al.* [129] et Amorim *et al.* [22], le fait d'inclure les setups a un impact direct sur l'utilisation de la capacité, et augmente considérablement la difficulté de résolution du modèle. Les auteurs classent les structures de setups en *setups simples* et *setups complexes*, ces derniers étant sous-divisés en trois catégories, auxquelles nous ajoutons les setups débordants, étudiés dans des travaux récents (voir par exemple [161] etr [211]).

- **Setup simple :** indépendant de la séquence et des décisions des périodes précédentes.

- **Setup complexe :**
  - **Setup reportable** (*setup carryover*) : si le produit/opération à traiter sur une machine au début de la période l est le même que celui traité sur la même machine à la fin de la période $l-1$, il n'y a pas de setup.
  - **Setup par famille** (*family setup*) : le coût et le temps de setup entre deux produits de la même famille (*minor setup*) sont plus petits que ceux entre deux produits de familles différentes (*major setup*).
  - **Setup débordant** (*setup crossover*, *period overlapping setup* ou *setup splitting*) : un setup peut être interrompu à la fin d'une période et repris au début de la période suivante, sans générer des coûts ni des temps additionnels.
  - **Setup dépendant de la séquence** : le coût ou le temps de setup générés par un produit sur une machine varient en fonction du produit précédemment traité.

Un exemple illustrant les setups complexes est présenté dans la Figure 2.2, où A, B, C, D correspondent à des produits/opérations à traiter sur une machine, sur un horizon de planification de 5 périodes. Les blocs restants correspondent aux durées de lancement (temps de setup). Nous pouvons clairement identifier un setup reportable, un setup débordant et un setup mineur (par famille). De plus, en comparant le temps de setup pour traiter A dans les périodes 4 et 5, nous observons qu'il s'agit dans ce cas de setups dépendants de la séquence, car le temps de setup de la machine pour produire A varie en fonction du produit précédemment traité (B à la période 3 et C à la période 4). Dans le cas d'un setup simple, les deux durées de lancement seraient identiques.

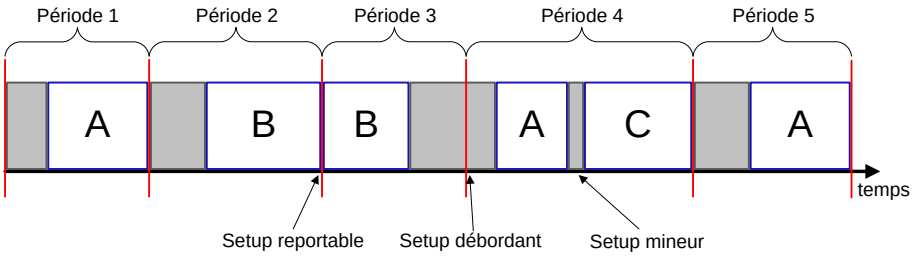

FIGURE 2.2 – Exemple de structures de setups complexes

Les problèmes de dimensionnement de lots peuvent être classés selon différentes caractéristiques. Les caractéristiques les plus utilisées dans la littérature sont : la capacité de production, le comportement de la demande, la longueur des périodes, le nombre de machines, le nombre de produits, le nombre de sites de production et le nombre de niveaux de nomenclature. Un état de l'art sur les problèmes de dimensionnement à un seul produit peut être consulté dans [33]. À partir de la nomenclature, les problèmes de dimensionnement de lots sont classés en : problèmes à un seul niveau et problèmes à plusieurs niveaux. Comme expliqué dans la Section 1.3 du Chapitre 1, le premier groupe correspond aux systèmes de production dans lesquels il n'y a pas de liens entre les produits, c'est-à-dire qu'aucun produit n'est composant d'un autre. En revanche, dans les problèmes multi-niveaux, certains produits sont des composants et d'autres des composés. On parle donc, dans ce dernier cas, de systèmes d'assemblage et de systèmes généraux. Les figures 2.3 et 2.4 présentent une classification des problèmes de dimensionnement de lots à un et plusieurs niveaux, respectivement.

Les problèmes de dimensionnement de lots avec une nomenclature à un seul niveau ont été nettement plus étudiés que les problèmes à plusieurs niveaux. L'avantage des problèmes

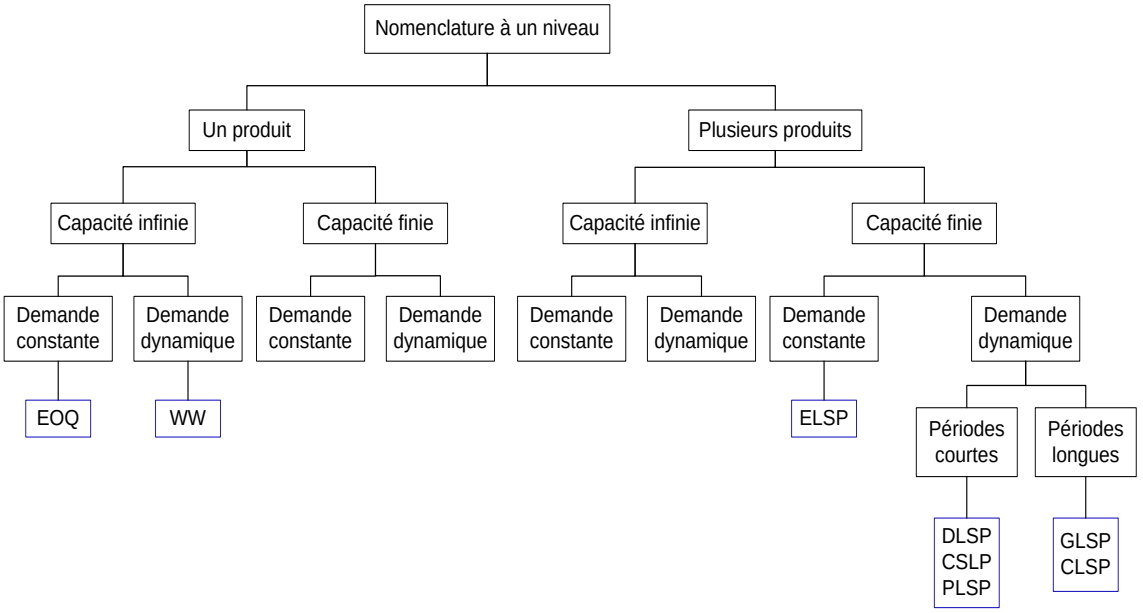

FIGURE 2.3 – Classification des problèmes de dimensionnement de lots à un seul niveau [233]

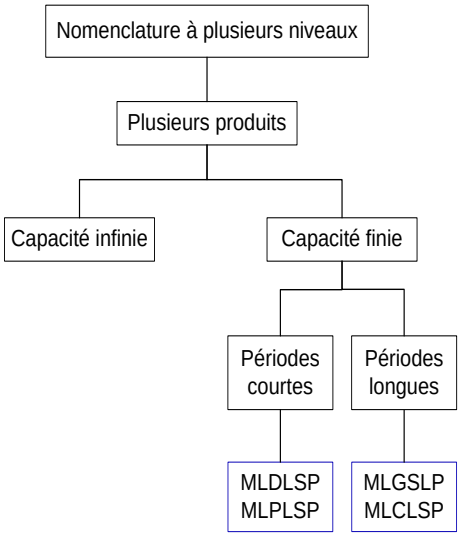

FIGURE 2.4 – Classification des problèmes de dimensionnement de lots à plusieurs niveaux [233]

mono-niveau, du point de vue de l'effort de calcul, est que leur modélisation comporte moins de variables de décisions (et parfois moins de contraintes) et moins de liens entre les variables de décision. Néanmoins, les systèmes de production associés à de tels modèles sont limités. En effet, dans les problèmes mono-niveau, les demandes de chaque produit sont indépendantes de la fabrication des autres produits, car les produits ne sont pas liés entre eux, i.e. aucun produit n'est

composant d'un autre. Cependant, même sous ces hypothèses, la résolution des problèmes de dimensionnement de lots avec contraintes de capacité à un seul niveau est souvent $\mathcal{NP}$-difficile. La difficulté de résolution est en particulier associée aux contraintes de capacité, et varie selon la configuration de l'atelier de production. Ainsi, les problèmes à une seule ressource sont plus faciles à résoudre que ceux à ressources multiples, car la capacité exacte du système est connue, les contraintes de séquencement détaillées ne sont pas indispensables et les modèles comportent moins de variables binaires. En effet, il n'y a pas de décisions d'affectation des opérations aux ressources à considérer ni de temps d'attente (la machine est toujours occupée), ce qui n'est plus vrai dans le cas multi-ressources. Les problèmes mono-ressource sont de loin les plus étudiés dans la littérature. D'ailleurs, les versions standard des modèles de dimensionnement de lots correspondent à des systèmes de production à une seule machine. Récemment, des extensions à des cas multi-ressources, principalement à des ateliers à machines en parallèle, ont été étudiées. Néanmoins, les hypothèses considérées dans la plupart de ces modèles ne sont applicables à des systèmes plus complexes, comme les ateliers de type flow-shop et job-shop, dont la modélisation nécessite des contraintes d'ordonnancement ou de capacité détaillée. Dans les problèmes de dimensionnement de lots avec contraintes de capacité, des contraintes d'ordonnancement typiquement considérées sont celles modélisant les temps et/ou coûts de setup dépendants des séquences. Un état de l'art sur les problèmes avec setups dépendants des séquences peut être consulté dans [244]. Bien que ces contraintes sont parfois importantes en pratique, elles ne suffisent en général pas pour garantir la faisabilité du plan de production au niveau ordonnancement.

Selon la longueur des périodes, les problèmes de dimensionnement de lots avec contraintes de capacité sont classés dans la littérature en problèmes à courtes périodes et problèmes à longues périodes. Nous nous intéressons, dans le reste de ce chapitre aux problèmes dans lesquels la demande est dynamique, c'est-à-dire varie dans le temps. Il existe tout de même le problème de dimensionnement de lots avec contraintes de capacité et demande constante, appelé *Economic Lot Scheduling Problem* (ELSP), qui est une extension du modèle EOQ au cas multi-produits et est $\mathcal{NP}$-difficile [115]. Le lecteur intéressé par ce problème peut consulter [187] et [103].

Nous discutons, dans ce qui suit, les principaux travaux scientifiques traitant les problèmes avec contraintes de capacité et à demande dynamique. Nous suivons la classification par longueur des périodes (problèmes à courtes et à longues périodes), en différentiant les travaux par type de nomenclature (mono-niveau et multi-niveaux) et par configuration de l'atelier de production (mono-ressource et multi-ressources).

2.3.1 Problèmes à courtes périodes

Les problèmes de dimensionnement de lots à courtes périodes sont caractérisés par un horizon de planification avec des périodes de courte durée (par exemple des heures ou des jours). D'autres caractéristiques ou hypothèses sont les suivantes :

- Atelier à une seule ressource,
- Limitations dans le nombre de lancements de production par période,
- Limitations dans l'utilisation de la capacité,
- Toutes les périodes ont la même longueur,
- Horizon de planification fini,
- Ordonnancement période par période.

Des généralisations des problèmes à courtes périodes pour des ateliers à plusieurs ressources ont aussi été proposées dans la littérature, principalement pour le cas de machines en parallèle.

Pour éviter d'avoir à décider du séquencement et de l'ordonnancement des opérations, des périodes de courtes durées et des limites sur le nombre de setups par périodes sont fixées. De cette façon, les produits ne sont pas en « compétition » sur l'utilisation des ressources, et le plan de production respecte la capacité. À cause de ces limitations, le nombre de variables binaires devient très important et, par conséquent, seuls quelques problèmes de petite taille (systèmes à une ou deux ressources et peu de produits) peuvent être résolus en des temps de calcul raisonnables. Ces problèmes sont en fait $\mathcal{NP}$-difficiles.

Le groupe de problèmes à courtes périodes est composé des problèmes ci-dessous :

- Discrete Lot-sizing and Scheduling Problem (DLSP),
- Continuous Setup Lot-sizing Problem (CSLP),
- Proportional Lot-sizing and Scheduling Problem (PLSP).

2.3.1.1 Discrete Lot-sizing and Scheduling Problem

La principale caractéristique de ce problème est qu'au maximum un produit peut être fabriqué par période (i.e. un setup par période). De plus, toute la capacité de la période est utilisée par la production de l'article si l'on décide de le produire. C'est ce que l'on appelle une politique de production de tout ou rien (« all or nothing »). L'horizon de planification est divisé en plusieurs périodes de même longueur. En ce qui concerne la capacité, il y a une contrainte par machine à chaque période. Un état de l'art sur le DLSP peut être consulté dans [70]. La version multi-niveaux du DLSP est connue sous l'appellation « Multi-Level Discrete Lot-sizing and Scheduling Problem » (MLDLSP).

Plusieurs variantes du DLSP on été étudiées dans la littérature. Pour plus de détails, le lecteur peut consulter le travail de Salomon *et al.* [198], qui décomposent le problème selon :

- Le comportement de la ligne de production,
- Le nombre de machines,
- Le nombre de produits,
- La structure du coût de setup,
- La structure du coût de production,
- La structure du temps de setup.

Les auteurs analysent la complexité des différentes variantes, lorsque l'on veut déterminer soit une solution réalisable, soit une solution optimale. Comme conclusion, trouver la solution optimale est $\mathcal{NP}$-difficile dans tous les cas, sauf pour le problème à machines parallèles identiques sans coûts de setup, qui est résolu de façon optimale en temps polynomial. Une autre étude sur la complexité du DLSP est présentée par Brüggemann et Jahnke [34], qui corrigent certains résultats trouvés dans [198], concernant les preuves de complexité.

Plusieurs formulations et méthodes de résolution basées sur des modèles MIP ont été proposées pour résoudre le DLSP à une seule machine. Salomon *et al.* [199] développent une méthode exacte pour résoudre le problème avec coûts et temps de setup dépendants des séquences, dénoté DLSPSD. Ils transforment le modèle mathématique en un problème de *voyageur de commerce* ([23]) avec fenêtres de temps. Gicquel *et al.* [79] proposent un modèle MIP pour le problème avec contraintes de coûts et de temps de setups dépendant des séquences. Ils formulent aussi le problème comme celui du voyageur de commerce avec fenêtres de temps et utilisent un algorithme de programmation dynamique pour le résoudre. Les auteurs montrent la similarité du problème avec le problème de séquencement de batchs. Gicquel *et al.* [78] proposent une nouvelle formulation du problème avec coûts de changement dépendant des séquences, en utilisant une structure de produit multi-attributs, un attribut étant une caractéristique physique (par exemple, la couleur, les dimensions, etc.). Les temps de changements ne sont pas considérés. La structure multi-attributs permet de rassembler les produits partageant certains attributs, de façon à utiliser des temps de changement par groupe, ce qui permet de diminuer le nombre de variables du modèle mathématique. Ils utilisent l'approche de Wolsey [237] pour obtenir des inégalités valides permettant de renforcer la modélisation. Des expérimentations sont menées pour comparer la performance des deux approches.

En ce qui concerne les systèmes à plusieurs ressources, Jans et Degraeve [121] proposent un modèle et une procédure de résolution pour le DLSP à machines parallèles chez un fabricant international de pneus. Le problème est décomposé par programmation dynamique en un programme maître, qui minimise le coût total, et un ensemble de sous-problèmes par pneu.

L'approche de résolution est basée sur un algorithme de génération de colonnes. Un travail récent pour un problème avec des machines parallèles identiques a aussi été proposé par Gicquel *et al.* [80]. Les auteurs changent la formulation standard, en remplaçant les variables individuelles associées à chaque machine par des variables agrégées, et ils comparent les formulations à l'aide d'un solveur commercial.

Pour ce qui est du cas multi-niveaux, le MLDLSP a rarement été étudié dans la littérature. Un des seuls travaux est celui de Jordan et Koppelmann [123], qui étudie le problème à une seule machine et propose deux façons de modéliser le problème. Le premier modèle correspond au MLDLSP standard (généralisation du DLSP) et le deuxième correspond à un problème de séquencement de batchs (MLBSP ou Multi-Level Batch Sequencing Problem) [122]. Le problème est résolu, en utilisant les deux modèles. Une heuristique de regrets aléatoires [133] est implémentée pour résoudre le modèle MLDLSP, et un *branch and bound* est utilisé pour résoudre le modèle MLBSP. En comparant le temps d'exécution et la qualité des solutions, les auteurs concluent que modéliser le problème comme un MLBSP offre une meilleure performance.

2.3.1.2 Continuous Setup Lot-sizing Problem

Le principe du CSLP est le même que celui du DLSP (on peut produire au maximum un article par période), sauf qu'il n'est pas obligatoire d'utiliser toute la capacité de production de la période pour produire un article (i.e. l'hypothèse « all or nothing » n'est pas considérée). Cette stratégie permet donc de diminuer les tailles des lots, et par conséquent de réduire les niveaux de stock. Donc, les solutions obtenues sont potentiellement meilleures que celles du DLSP. Ce problème, qui est $\mathcal{NP}$-difficile [73], a aussi été appelé « Deterministic Dynamic Product Cycling Problem ». De manière générale, le CSLP n'a pas généré beaucoup d'intérêt dans la littérature, et à notre connaissance, il n'a jamais été étendu au cas multi-niveaux.

Karmarkar et Schrage [131] décrivent plusieurs formulations utilisées pour modéliser le CSLP à une seule machine sans coûts de setup, et utilisent une relaxation Lagrangienne pour calculer des bornes inférieures, qui sont ensuite utilisées dans un algorithme de *branch and bound*.

Karmarkar *et al.* [130] présentent un modèle mathématique pour le problème à une seule machine avec coûts de démarrage des machines (*startup costs*) et coûts de maintien de la machine en fonctionnement (*reservation costs*). Ils résolvent le problème sans contraintes de capacité par un algorithme de programmation dynamique, et utilisent une procédure de *branch and bound* avec une relaxation Lagrangienne pour résoudre le problème avec contraintes de capacité. Le même problème est résolu par Hindi [112], au moyen d'une recherche taboue. D'autre part, Sandbothe [200] propose une heuristique dédiée pour le problème avec coûts

de setup. Vanderbeck [227] étudie le problème à une seule machine avec coûts de setup et de démarrage de la machine. Il résout le problème à un seul produit avec un algorithme de génération de colonnes, et le problème à plusieurs produits par une procédure de programmation dynamique.

Constantino [42] propose des inégalités valides pour rendre la formulation à une seule machine plus robuste et permettre d'obtenir des bornes inférieures (par relaxation linéaire) de meilleure qualité. Finalement, il résout le problème au moyen d'un algorithme de *branch and cut*.

Almada-Lobo *et al.* [18] ont récemment étudié le CSLP avec coûts et temps de setup dépendant des séquences, dans un atelier multi-ressources. Dans leur travail, les auteurs proposent deux modèles de programmation linéaire en variables mixtes avec des inégalités valides. De Matta et Guignard [50] étudient aussi un problème multi-ressources, mais les temps de setup ne sont pas considérés.

2.3.1.3 Proportional Lot-sizing and Scheduling Problem

Le PLSP est une généralisation (ou variante) du CSLP, dans la mesure où la capacité restante à chaque période peut être utilisée pour fabriquer un deuxième produit, sous la condition que les deux produits soit traités sur la même ressource. Cependant, un seul setup est autorisé par période, c'est-à-dire que le deuxième produit à fabriquer dans la période t est le premier à la période $t + 1$. Le PLSP, dont des extensions sont étudiées par Drexl et Haase [60], permet d'obtenir des meilleures solutions (en termes de coûts) que le CSLP et le DLSP, en utilisant mieux la capacité des périodes [136]. Ce problème a aussi été étudié pour le cas multi-niveaux, et il est nommé « Multi-Level Proportional Lot-sizing and Scheduling Problem » (MLPLSP).

Drexl et Haase [60] présentent un modèle mathématique et une heuristique de regrets aléatoires (*randomized-regret-based biased sampling method*) pour le problème mono-niveau à une seule machine. Suerie [211] présente deux nouveaux modèles, en considérant des setups débordants.

En ce qui concerne les ateliers multi-ressources, Drexl et Haase [60] présentent un modèle mathématique standard. Une extension de cette formulation, en considérant un opérateur commun pour réaliser les opérations de setup est présentée dans [214]. Le problème avec machines parallèles identiques a été étudié par Kaczmarczyk [127], qui propose plusieurs modélisations, en remplaçant les contraintes binaires par des variables entières, permettant d'agréger les machines selon certaines conditions (par exemple, le nombre de machines disponibles pour traiter un produit particulier).

Pour ce qui est des problèmes multi-niveaux, Kimms [132] propose deux heuristiques pour résoudre le MLPLSP mono-ressource. La première est basée sur des regrets aléatoires, comme dans [60], et la deuxième est une recherche taboue qui utilise la notion de graphe disjonctif. Les deux heuristiques construisent les plans de production d'avant en arrière, en affectant au maximum deux produits à chaque période. Stadtler [208] présente un modèle avec des délais d'obtention nuls et un deuxième modèle avec temps de setups débordants et des contraintes de tailles de batchs (groupe de produits de la même famille). Il résout un problème de l'industrie pharmaceutique au moyen d'un solveur standard.

Le MLPLSP à plusieurs machines a surtout été traité à travers l'étude de modèles mathématiques, permettant de tenir compte de différentes configurations d'ateliers de production. Un travail de base important est celui de Kimms et Drexl [136], où les auteurs discutent des principales caractéristiques du PLSP, en mettant l'accent sur le MLPLSP multi-ressources, pour lequel un modèle MIP est expliqué en détail. Puis, dans [135], les mêmes auteurs présentent différentes formulations MIP pour le MLPLSP multi-ressources, en faisant varier l'organisation des machines dans l'atelier de production (machines en parallèle, plusieurs machines par produit et machines partiellement renouvelables). Une méthode de résolution basée sur un algorithme génétique est proposée par Kimms [134], qui présente aussi une formulation MIP du problème à plusieurs ressources. Des résultats expérimentaux valident l'intérêt de l'approche face à la méthode hiérarchique classique MRP-II.

2.3.2 Problèmes à longues périodes

Dans les problèmes à longues périodes, la durée des périodes est souvent de l'ordre de quelques semaines ou de quelques mois. La caractéristique générale est que plusieurs produits peuvent être fabriqués par période sur une seule machine. Des extensions à des cas multi-ressources ont aussi été proposées. Contrairement aux problèmes à courtes périodes, qui suivent plus ou moins les mêmes hypothèses, les problèmes à longues périodes diffèrent substantiellement. Les deux problèmes les plus connus sont :

- Le General Lot-sizing and Scheduling Problem (GLSP),
- Et le Capacitated Lot-Sizing Problem (CLSP).

Le GLSP et le CLSP sont des problèmes $\mathcal{NP}$-difficiles, et ont beaucoup été étudiés car les limitations trouvées dans les problèmes à courtes périodes n'existent plus ou sont moins importantes. Le CLSP est le problème de dimensionnement de lots avec contraintes de capacité le plus étudié dans la littérature, et plusieurs états de l'art sont disponibles ([129], [189], [36]).

2.3.2.1 General Lot-sizing and Scheduling Problem

Dans le GLSP, l'horizon de planification est divisé en un certain nombre de macro-périodes (longues périodes), et chaque macro période est divisée en quelques micro périodes (courtes périodes) de longueur variable. Dans chaque micro-période, il est possible de produire uniquement un article, en utilisant l'hypothèse « all or nothing », comme dans le DLSP. Cependant, ce problème, qui est aussi $\mathcal{NP}$-difficile [71], requiert encore plus d'effort de calcul que le DLSP, car non seulement les tailles de lots sont déterminées, mais aussi la séquence des lots par macro période. La longueur de chaque micro période est une décision indirecte qui dépend de la taille de lot du produit associé. Dans la version standard, le nombre de micro périodes par macro période est fixe. Une généralisation du GLSP est sa version multi-niveaux, appelée « Multi-Level General Lot-sizing and Scheduling Problem » (MLGLSP).

Plusieurs procédures basées sur des méthodes de recherche locale ont été proposées pour résoudre le GLSP à une seule machine. Fleischmann et Meyr [71] présentent trois heuristiques de recherche locale basées sur la méthode *threshold accepting*. Meyr [162] propose une approche qui combine un algorithme de réoptimisation du problème dual avec la méthode *threshold accepting* et avec un recuit simulé, pour résoudre le problème avec temps de setup dépendant des séquences.

Le problème avec temps et coûts de setup dépendants des séquences a été étudié par Araujo *et al.* [24]. Les auteurs proposent un modèle MIP, avec prise en compte de la rupture de stock, pour un atelier mono-machine dans une fonderie, et ils résolvent le problème par une heuristique de *relax and fix*. L'heuristique est améliorée, en utilisant trois variantes de recherche locale (une heuristique de descente, une recherche de voisinage diminué et un recuit simulé). L'approche suit une structure d'horizon glissant, où seuls les jobs immédiats sont ordonnancés. Un autre cas industriel, modélisé comme un GLSP avec temps et coûts de setup dépendants des séquences, a été étudié par Ferreira *et al.* [68]. Dans leur travail, les auteurs proposent un modèle MIP, une approche de relaxation et plusieurs heuristiques de *relax and fix* pour résoudre le problème de production de boissons. Tiacci et Saetta [220] étudient aussi le problème avec temps et coûts de setup dépendant des séquences, pour lequel ils présentent trois formulations MIP, qui correspondent à des versions simplifiées du modèle détaillé. Les modèles sont basés sur une stratégie d'horizon glissant, où les séquences d'ordonnancement sont considérées uniquement pour la première période. Dans le premier modèle, les contraintes de coûts de setup dépendant des séquences sont considérées uniquement pour la première période ; le deuxième modèle inclut un terme approximatif associé aux futurs coûts de setup, sans détailler la séquence, et le troisième modèle inclut une contrainte additionnelle pour éviter de produire un article à la première période, si son inventaire initial est supérieur aux besoins de la première période. Ferreira *et al.*

[67] étudient un système de production de boissons constitué de deux étages de production. Ils proposent quatre formulations mono-étage basées, pour les deux premières sur le GLSP avec temps et coûts de setup dépendant des séquences, et pour les deux autres sur le problème du voyageur de commerce. Cette dernière stratégie donne des meilleurs résultats.

Toso *et al.* [222] étudient un système de production de produits alimentaires pour animaux. Ils développent deux variantes de la méthode de *relax and fix* et utilisent des formulations alternatives pour résoudre le problème. Les stratégies déployées sont plus avantageuses que la résolution exacte d'un modèle MIP général.

Un problème multi-ressources à été traité par Meyr [163], qui s'intéresse à un atelier à machines parallèles non identiques. L'auteur propose un modèle mathématique et une procédure de résolution qui combine un algorithme de recherche locale et un recuit simulé.

En ce qui concerne le cas à plusieurs niveaux, un des travaux modélisant le MLGLSP est celui de Fandel et Stammen-Hegene [64]. Les auteurs étudient le problème à plusieurs machines, généralisé pour une configuration de type job-shop, avec coûts de setup dépendants des séquences. La formulation proposée permet d'obtenir des solutions réalisables au niveau opérationnel pour des systèmes complexes. Un inconvénient, cependant, est que les délais d'obtention (ou livraison) des produits intermédiaires de la nomenclature ne sont pas considérés. Du point de vue de l'effort de calcul, l'inconvénient majeur est que le modèle mathématique comporte beaucoup de variables avec trois différents types de variables binaires, ce qui rend très difficile la résolution du problème. En effet, comme expliqué par les auteurs, seulement des problèmes de très petite taille (avec peu de produits et peu de périodes) peuvent être résolus. Un autre travail sur le MLGLSP à plusieurs machines est celui de Seeanner et Meyr [202], qui présentent un modèle MIP pour le problème avec coûts de setup dépendant des séquences, coûts de temps morts (inactivité), coûts d'achat externe, coûts de temps additionnel et coûts de stockage de produits semi-finis. Comme le modèle est très difficile à résoudre en un temps raisonnable, même pour des petites instances, les auteurs proposent et comparent plusieurs reformulations du problème, en utilisant trois heuristiques : *truncated MIP*, *LP and fix* et *relax and fix*. La première heuristique offre une meilleure performance, mais le problème reste soluble seulement pour des petites instances. Le même problème est étudié par Seeanner *et al.* [201]. Les auteurs proposent une heuristique combinant les principes de la *recherche de décomposition de voisinage variable* (VNDS) [105], qui est une variante de la métaheuristique VNS, et de la méthode *fix and optimize*. La performance de l'heuristique est comparée avec celle d'un solveur commercial, montrant la pertinence de la méthode, surtout lorsque la taille des problèmes augmente.

2.3.2.2 Capacitated Lot-Sizing Problem

Dans le CLSP il n'y a pas de limitations sur le nombre de setups par période, les périodes ne sont pas divisées en sous-périodes, et n'importe quelle fraction de la capacité disponible par période peut être utilisée. Bitran et Yanasse [31] ont montré que ce problème était $\mathcal{NP}$-difficile. Parmi les problèmes de dimensionnement de lots avec contraintes de capacité, le CLSP est le problème le plus étudié, car il tient compte de la capacité exacte et que le modèle mathématique est moins complexe. Le CLSP à plusieurs niveaux est nommé dans la littérature « Multi-Level Capacitated Lot-sizing Problem » (MLCLSP). Des états de l'art focalisés sur le CLSP sont réalisés dans [129], [189] et [36].

Comme dans les autres problèmes, le modèle mathématique standard du CLSP ne garantit des solutions réalisables que pour des systèmes de production mono-ressource. Le CLSP à une seule machine ne nécessite pas de décisions de séquencement, à moins que les temps et coûts de lancement de la production varient en fonction des séquences. Différentes extensions au cas multi-ressources ont aussi été proposées dans la littérature. La formulation du CLSP multi-ressources impose la considération des dates de début et de fin des opérations dans les contraintes de capacité, et rend le problème beaucoup plus difficile à résoudre. C'est pourquoi les problèmes multi-ressources étudiés restent typiquement associés aux ateliers à machines parallèles. Les ateliers de type flow-shop et de type job-shop sont très peu étudiés.

Comme pour les autres problèmes, les contraintes d'ordonnancement généralement étudiées dans le CLSP sont celles associées aux temps et/ou aux coûts de setup dépendant des séquences. Contrairement au GLSP, où ces contraintes sont inhérentes au problème (la séquence fait partie des décisions du modèle standard), dans le CLSP, ces contraintes créent une difficulté additionnelle très importante, surtout dans les cas de systèmes avec partage de ressources et à séquencement variable, comme les ateliers de type job-shop, car il n'y a pas de limites sur le nombre de setups par période.

Le CLSP à une seule machine avec des coûts de setup dépendant des séquences, noté CLSD, a été notamment introduit par Haase [100], sans tenir compte des temps de setup. Une méthode de résolution pour ce problème a été proposée par Shim *et al.* [205], en utilisant une heuristique à deux niveaux. Le problème avec temps et coûts de setup dépendants des séquences a été étudié par Haase et Kimms [102]. Dans leur travail, les auteurs présentent un modèle MIP considérant uniquement les séquences efficaces et utilisent un algorithme de *branch and bound* pour résoudre le problème. Une séquence est efficace s'il n'existe pas une autre séquence des mêmes produits avec un coût de setup inférieur. Almada-Lobo *et al.* [17] proposent un nouveau modèle pour le même problème. Kovács *et al.* [140] présentent aussi un autre modèle, et proposent une heuris-

tique pour résoudre des instances de grande taille avec des temps de calcul raisonnables. Gupta et Magnusson [99] étudient le même problème, mais avec setups débordants. Ils présentent une formulation MIP et proposent une heuristique de résolution. Les résultats sont meilleurs quand le nombre de produits est significativement plus grand que le nombre de périodes de l'horizon de planification. Néanmoins, Almada-Lobo *et al.* [19] montrent que ce modèle n'empêche pas la génération de sous-tours déconnectés, et ne garantit donc pas l'obtention de solutions réalisables. Pour éviter ce problème, les auteurs proposent des nouvelles contraintes à ajouter dans le modèle original. Une nouvelle formulation, incluant des setups débordants et dépendant des séquences, est présentée par Menezes *et al.* [161]. Les auteurs comparent ce modèle avec le GLSP, et ils prouvent que leur formulation donne des meilleurs résultats. Almada-Lobo et James [16] étudient aussi le problème avec temps et coûts de setup dépendant des séquences, et proposent deux heuristiques pour résoudre le problème : une recherche taboue et une recherche avec voisinage variable. La recherche taboue a aussi été mise en œuvre par Hindi [113], et la recherche à voisinage variable a été également utilisée par Almada-Lobo *et al.* [20] pour un problème de l'industrie du verre.

Une technique très utilisée pour résoudre le CLSP est la relaxation Lagrangienne. Ainsi, Thizy et Van Wassenhove [219] proposent une heuristique Lagrangienne basée sur la relaxation des contraintes de capacité et utilisant la méthode des sous-gradients. Diaby *et al.* [56] proposent une formulation MIP et des méthodes de résolution basées sur la relaxation Lagrangienne. Les résultats expérimentaux présentent des écarts par rapport à l'optimum inférieurs à 1% avec des temps de calculs petits. Diaby *et al.* [55] proposent une heuristique Lagrangienne pour résoudre des problèmes de grande taille. Millar et Yang [164] proposent deux procédures pour résoudre le CLSP avec rupture de stock, formulé comme un problème de transport : une décomposition Lagrangienne [97] et une relaxation Lagrangienne. Trigeiro *et al.* [223] étudient le CLSP avec temps de setup. Ils présentent une heuristique Lagrangienne, où les contraintes de capacité sont relâchées. Pour construire des solutions réalisables, une heuristique de lissage est implémentée.

Absi et Kedad-Sidhoum [4] étudient le problème avec des coûts de pénurie, qui sont considérés quand la demande est partiellement ou complètement perdue. Ils proposent des inégalités valides et présentent des algorithmes de coupes pour la mise en œuvre de ces inégalités.

Tempelmeier [213] étudie le problème avec demande aléatoire (SCLSP, Stochastic CLSP), où les demandes non satisfaites induisent des coûts de rupture. Une procédure combinant un algorithme de génération de colonnes et l'heuristique ABC_β (présentée dans [217]) est proposée. Helber *et al.* [111] étudient également le SCLSP, mais en incluant des stocks de sécurité. La formulation exacte du problème étant non linéaire, les auteurs proposent deux approximations

sous forme de modèles linéaires. Une heuristique de *fix and optimize* est utilisée pour résoudre les deux modèles.

Goren *et al.* [94] présentent une approche hybride combinant un algorithme génétique avec une heuristique de *fix and optimize* pour résoudre le CLSP avec setups reportables. Le même problème, en ajoutant des setups débordants a été étudié par Mohan *et al.* [170], qui présentent un modèle MIP et réalisent des expérimentations montrant l'importance de considérer cette structure de setup.

En ce qui concerne les problèmes multi-ressources, plusieurs travaux traitant le CLSP avec machines parallèles ont été étudiés dans la littérature (voir par exemple [221] et [156]). Cependant, l'un des seuls travaux à inclure des contraintes d'ordonnancement est celui de Mateus *et al.* [158]. Les auteurs étudient le problème avec des coûts et des temps de setup dépendant des séquences dans un atelier avec machines parallèles non reliées, et ils proposent un modèle mathématique avec ordonnancement détaillé. D'abord, le problème de dimensionnement de lots est résolu de manière exacte, en considérant la capacité agrégée, et le problème d'ordonnancement est ensuite résolu de manière sous-optimale, en utilisant une heuristique GRASP. Un inconvénient est que le problème d'ordonnancement est résolu période par période et, par conséquent, quand la charge de travail dépasse la capacité d'une période, des ruptures de stock sont considérées. Si l'ordonnancement était réalisé sur tout l'horizon de planification, il serait possible d'éviter la rupture de stocks dans plusieurs cas. James et Almada-Lobo [120] proposent une approche combinant plusieurs heuristiques basées sur un modèle MIP (extension du CLSP et du CLSD) pour résoudre le problème à plusieurs machines en parallèle, avec coûts et temps de setups dépendant des séquences. L'algorithme comporte une recherche locale et une étape de modification du voisinage. Ils mettent en évidence que les instances avec coûts de setup élevés sont plus difficiles à résoudre pour toutes les heuristiques. Le problème original est décomposé en plusieurs sous-problèmes pouvant être plus faciles à résoudre. Certaines variables de décision sont fixées à chaque itération. Une première solution est obtenue à l'aide d'une procédure de *relax and fix*. Mohammadi [167] présente un modèle mathématique pour un problème de planification, de chargement et d'ordonnancement avec setups dépendant des séquences et avec une configuration d'atelier de type flow-shop flexible. L'auteur propose des heuristiques basées sur la programmation mathématique pour la résolution du problème et réalise une comparaison avec les solutions optimales obtenues par un solveur. Mohammadi *et al.* [169] étudient le CLSP avec setups dépendants des séquences dans un flow-shop. Un modèle MIP et cinq heuristiques de résolution sont présentés. Les heuristiques sont basées sur des versions simplifiées du modèle et chacune est adaptée pour une taille d'instance différente.

Concernant les nomenclatures à plusieurs niveaux, le MLCLSP est principalement étudié dans le cadre d'ateliers mono-ressource, pour lesquels différents modèles et méthodes de résolutions ont été proposées. Une analyse comparative sur l'efficacité des modèles mathématiques à une seule machine est présentée dans [238]. Hung et Chien [116] proposent un modèle mathématique pour différents types de demandes (demandes en rupture, demandes confirmées, stock de sécurité, prévisions fiables et prévisions incertaines). Ils résolvent différentes instances du problème, en utilisant trois métaheuristiques : une recherche taboue, un recuit simulé et un algorithme génétique. Les résultats sont meilleurs avec la recherche taboue quand la demande suit le modèle des *demandes confirmées*, et le recuit simulé et plus avantageux quand on travaille avec des prévisions. Mohammadi *et al.* [168] proposent deux modèles MIP avec setups dépendant des séquences. Ils proposent aussi quatre heuristiques (deux de type *horizon glissant* et les deux autres de type *fix and relax*). Selon la taille des instances, une heuristique peut être plus favorable que d'autres. Sahling *et al.* [197] proposent une heuristique de type *fix and optimize* [185] pour résoudre le problème avec setups reportables. Almeder [21] propose une approche d'optimisation hybride combinant un algorithme de colonies de fourmis avec une méthode exacte utilisant un solveur standard. Helber et Sahling [110] proposent une approche de *fix and optimize* pour résoudre le MLCLSP avec délais d'obtention positifs, où un ensemble de sous-problèmes sont résolus avec un nombre limité de variables binaires, et Wu *et al.* [239] proposent deux formulations MIP avec prise en compte des ruptures de stock, ainsi qu'une méthode d'optimisation dont les bornes inférieures obtenues à partir de relaxations du modèle sont de bonne qualité.

Le MLCLSP à plusieurs ressources a été rarement étudié. Un des seuls travaux à s'intéresser à ce problème est celui d'Akartunalı et Miller [14], qui présentent une méthode pouvant être adaptée à plusieurs types de problèmes de planification à longues périodes. Ils proposent un modèle mathématique standard (extension du CLSP) et une formulation avec variables d'échelon stock [39]. Un inconvénient de ces modèles est que les contraintes de capacité considérées sont agrégées. La méthode de résolution proposée est basée sur des heuristiques de type *LP and fix* et *relax and fix*, et elle s'avère plus efficace qu'un solveur commercial.

2.4 Problèmes d'ordonnancement

L'objectif de ce travail de thèse n'est pas d'étudier en profondeur les problèmes d'ordonnancement. Nous sommes intéressés par l'intégration des décisions de planification et d'ordonnancement au niveau tactique. Notre problématique principale est par conséquent un problème de planification avec contraintes d'ordonnancement ou de capacité détaillées. Dans cette section,

nous proposons tout de même un bref aperçu des problématiques traitées par la communauté scientifique autour des problèmes d'ordonnancement.

Comme discuté dans le chapitre 1, les problèmes d'ordonnancement peuvent être classés selon la configuration de l'atelier de production, selon le critère d'optimisation et selon les contraintes. Un autre aspect permettant de classer ces problèmes est la considération ou pas des setups, dans une ou deux dimensions : temps et/ou coût. Un état de l'art suivant cette classification, et pour différents types d'ateliers de production, peut être consulté dans [15].

Dans la plupart des problèmes d'ordonnancement traités dans la littérature, les setups sont indépendants des séquences. D'une part, les temps de setup peuvent souvent être inclus dans les durées opératoires des tâches de production et, d'autre part, les coûts de setups ne sont pas considérés dans la fonction objectif, les critères d'optimisation étant la plupart du temps liés au temps de fin des tâches. Le critère le plus étudié est le *makespan*. Les contraintes de setups dépendant des séquences sont principalement considérées dans les problèmes intégrant le dimensionnement de lots et l'ordonnancement (voir par exemple [17]), mais les modèles considérés sont souvent à une machine.

Les problèmes d'ordonnancement peuvent aussi être classés en : ordonnancement avec batch [186] et ordonnancement sans batch. Un batch (conteneur, palette, etc.) étant formé par plusieurs jobs de la même famille, le but est d'ordonnancer les jobs par groupes, en économisant en coûts et en temps de production. Dans l'ordonnancement sans batch, chaque job est séquencé de manière indépendante.

De manière générale, les problèmes d'ordonnancement font partie des problèmes combinatoires les plus difficiles à résoudre. Hormis la considération ou pas de setups et de contraintes spécifiques, la complexité des problèmes d'ordonnancement dépend fortement du nombre de ressources et de jobs (i.e. de la configuration de l'atelier). La plupart des problèmes généraux sont $\mathcal{NP}$-difficiles, y compris une grande partie des problèmes à une seule machine [147], mais les problèmes les plus difficiles sont ceux associés aux ateliers de type job-shop, car le nombre de séquences possibles augmente de manière exponentielle quand le nombre de ressources ou de jobs augmente de manière linéaire. La difficulté est encore plus importante si le système est flexible (opérations pouvant être réalisées sur plusieurs machines), car non seulement l'ordre de traitement des jobs sur les machines doit être décidé, mais aussi l'affectation des jobs aux machines, ce qui augmente le nombre de variables binaires à considérer dans la modélisation.

Plusieurs types d'approches ont été développées pour la résolution des problèmes d'ordonnancement. Les méthodes exactes les plus utilisées sont le *branch and bound* et la programmation dynamique. Les métaheuristiques les plus appliquées sont la recherche taboue, la recherche à voisinage variable, le recuit simulé et les algorithmes génétiques.

Nous nous intéressons dans les chapitres qui suivent à l'ordonnancement sans batch et avec setups qui sont indépendants des séquences, dans un atelier de type job-shop. Un état de l'art sur les techniques utilisées pour la résolution du problème d'ordonnancement dans un atelier de type job-shop peut être consulté dans [119].

2.5 Intégration de la planification et de l'ordonnancement

L'intégration de la planification et de l'ordonnancement permet de garantir la faisabilité des plans de production décidés au niveau tactique. En traitant la planification et l'ordonnancement de manière séparée, on ne tient pas compte des contraintes de succession qui ont un impact sur la capacité exacte du système, et la détermination d'un plan de production réalisable au niveau opérationnel n'est pas garantie. Il est par conséquent important d'essayer de prendre les deux types de décision en même temps.

Comme expliqué dans la Section 1.4.1 du Chapitre 1, l'approche MRP permet de planifier la production sans tenir compte de la capacité du système. Quand le plan de production est construit, le but est d'ordonnancer au mieux les activités pour respecter le plan de production qui est supposé être optimal en termes de coût. Le problème apparaît lorsqu'il est impossible de déterminer un ordonnancement respectant les délais de production, dû à une capacité insuffisante du système de production. Dans ce cas, toutes les stratégies envisageables pour respecter le plan de production entraîneront une augmentation du coût global.

MRP-II prend en compte certaines informations sur la capacité de production avec différents niveaux de détails, qui sont désagrégés hiérarchiquement, en construisant plusieurs plans de production jusqu'à obtenir un plan détaillé. Cette procédure d'estimation de la capacité génère une vision limitée des ressources au niveau tactique et conduit vers le calcul d'un plan qui n'utilise pas de manière efficace la capacité, et qui peut être infaisable au niveau ordonnancement. D'où l'importance de suivre une approche qui intègre du mieux possible les contraintes du niveau opérationnel.

Les modèles mathématiques correspondant aux problèmes DLSP, CSLP, PLSP, GLSP, CLSP et ses différentes variantes garantissent des solutions réalisables au niveau opérationnel pour des systèmes de production relativement simples. C'est le cas des ateliers à une seule machine ou deux machines parallèles, ainsi que les systèmes à un seul produit ou plusieurs produits sans gamme de fabrication. Par contre, quand on considère des systèmes de production plus complexes avec différents produits ayant des gammes de fabrication et plusieurs ressources (par exemple, les ateliers de type flow-shop et job-shop), les modèles classiques de planification ne garantissent pas des solutions réalisables. En effet, pour que les plans de production soient

réalisables, il faut prendre en compte des contraintes de capacité détaillées qui incluent des contraintes d'ordonnancement. Il est notamment nécessaire de tenir compte des contraintes de précédence liées aux gammes et aux ressources, permettant de définir les dates de début et de fin des opérations. Les travaux présentés dans la section 2.2, bien qu'ils permettent de trouver des solutions réalisables et de bonne qualité pour des cas d'études particuliers, ne peuvent pas être considérés comme des approches intégrées (considérant des contraintes de capacité détaillée du niveau opérationnel), puisqu'une généralisation de ces méthodes ne garantit pas des solutions réalisables dans des ateliers de production complexes.

Par exemple, les contraintes de coûts et de temps de setup dépendant des séquences, très étudiées dans des travaux récents, sont des contraintes d'ordonnancement qui ne suffissent pas à garantir des solutions réalisables dans les systèmes multi-ressources complexes. Elles permettent uniquement de sélectionner la séquence qui satisfait le mieux un critère de coût ou de temps. Pour garantir des plans de production réalisables, les dates de début et de fin de toutes les opérations doivent être considérées dans le modèle mathématique.

Pour plus de détail sur l'importance d'intégrer les décisions d'ordonnancement dans la planification de la production au niveau tactique, le lecteur peut consulter le travail de Dauzère-Pérès et Lasserre [48], qui met en évidence les limites des approches hiérarchiques.

À ce sujet, Maravelias et Sung [155] présentent une classification des stratégies de résolution des problèmes de planification et d'ordonnancement, comme illustré dans la Figure 2.5. Ils définissent trois catégories de méthodes de résolution qui sont :

- Approches hiérarchiques,
- Approches itératives,
- Approches intégrées.

Dans les approches hiérarchiques, le problème de planification est résolu dans un premier temps, sans tenir compte des contraintes détaillées d'ordonnancement. Ainsi, le plan de production n'est pas forcément réalisable au niveau d'ordonnancement, et la solution correspond à une borne inférieure du problème intégré. Dans un deuxième temps, le problème d'ordonnancement est résolu, en fixant les variables de décisions déterminées dans le niveau de planification. Le problème se pose lorsqu'il devient impossible de trouver un ordonnancement réalisable pour suivre le plan de production. De plus, il est probable que l'écart entre la solution obtenue et l'optimum soit très grand.

Dans les approches itératives, il y a un échange systématique d'information entre le niveau résolvant le problème de planification (qui peut contenir ou non des contraintes approximant les contraintes réelles d'ordonnancement) et le niveau dédié à la résolution du problème d'or-

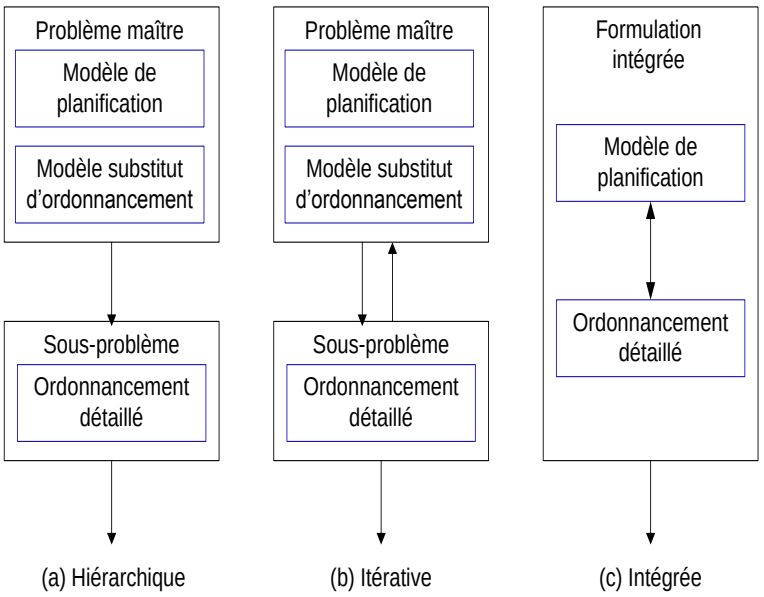

FIGURE 2.5 – Stratégies de résolution du problème intégré de planification et d'ordonnancement [155]

donnancement. Cette stratégie permet d'obtenir une solution réalisable, mais l'optimum global n'est pas garanti.

Dans les approches intégrées, il n'y a pas de séparation entre les problèmes de planification et d'ordonnancement, mais un seul problème est considéré avec une formulation intégrée. Résoudre un tel modèle permet d'obtenir la solution optimale du problème global, mais la complexité du problème étant très importante, il est souvent impossible d'obtenir une solution optimale dans des temps de calcul acceptables, voire même d'obtenir une solution réalisable. Seuls des problèmes de petite taille peuvent être résolus de façon optimale. C'est pourquoi il est nécessaire d'utiliser des méthodes de décomposition ou de relaxation, pour diminuer la complexité du problème dans les approches intégrées. L'optimalité de la solution n'est pas garantie, mais le rapport entre qualité de solution et temps de calcul devient acceptable.

Le but de cette section étant d'étudier les problèmes qui intègrent de manière cohérente les décisions de planification et d'ordonnancement, nous nous focalisons dans ce qui suit sur les approches itératives et les approches intégrées. Le lecteur intéressé par les méthodes hiérarchiques peut consulter [160] et [229].

2.5.1 Approches itératives

Approche de Lasserre [144]

Cette approche est la première à traiter de façon cohérente le problème mono-niveau de planification et d'ordonnancement de la production, et elle est la première méthode à garantir la faisabilité des plans de production au niveau opérationnel. L'approche suit une procédure itérative, où le problème intégré est décomposé en k problèmes de planification avec séquence fixée et k problèmes d'ordonnancement avec tailles de lots fixées, où k est le nombre d'itérations de la procédure de résolution. À chaque itération, un problème de planification et un problème d'ordonnancement sont résolus. Contrairement à d'autres travaux, qui traitent le problème d'ordonnancement période par période, l'ordonnancement est considéré sur tout l'horizon de planification comme un seul problème.

La méthode démarre avec le calcul d'une séquence initiale y, en résolvant le problème d'ordonnancement pour un plan de production dont les tailles de lot sont égales aux demandes. Ensuite, la séquence déterminée y est fixée et le problème de planification avec séquence fixée $P(y)$ est résolu. Puis, un nouveau problème d'ordonnancement (en fixant les tailles de lots calculées dans le plan précédent) est résolu, et un nouveau plan est déterminé. Cette procédure est répétée durant k itérations.

Le schéma général de la procédure itérative, qui converge vers un optimum local, est illustré dans la Figure 2.6.

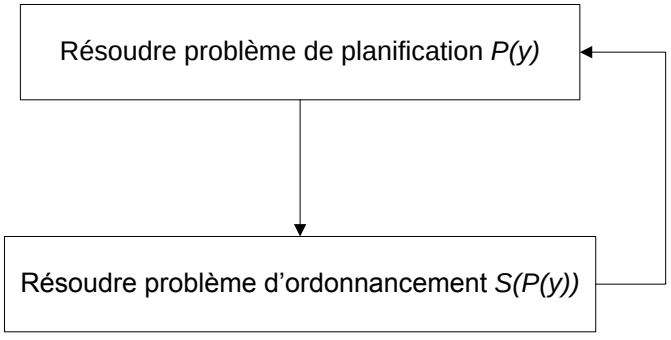

FIGURE 2.6 – Approche itérative de Lasserre [144]

Pour résoudre le problème de dimensionnement de lots à séquence fixée, l'auteur propose un modèle mathématique MIP qui est une extension du CLSP, en tenant compte des contraintes de capacité détaillées associées à la séquence fixée, information suffisante pour garantir la faisabilité du plan au niveau d'ordonnancement. Le modèle tient compte de la rupture des stocks et des

temps et des coûts de setup. Le problème d'ordonnancement est résolu à travers une version modifiée de l'*heuristique par machine goulot* (Shifting Bottleneck heuristic) [6], qui suit un critère de minimisation du *makespan*.

La performance de l'approche a été testée en considérant un atelier de type job-shop avec 6 produit, 6 machines et 3 périodes de planification. Différentes instances ont été générées à partir de plans initiaux différents. Dans la plupart des cas, la solution optimale ou une solution proche de l'optimum a été obtenue après peu d'itérations. Le problème de dimensionnement de lots avec séquence fixée ($P(y)$) a été résolu avec le solveur commercial XPRESS-MP [241].

Approche de Dauzère-Pérès et Lasserre [47]

Les auteurs proposent deux stratégies pour résoudre le problème d'ordonnancement, en suivant l'approche itérative de Lasserre [144]. Le problème d'ordonnancement est encore une fois résolu en utilisant une version modifiée ([46]) de l'heuristique par machine goulot [6], qui a pour but de minimiser le *makespan*.

Les stratégies de résolution du problème d'ordonnancement sont basées sur deux politiques : ordonnancement *global* et ordonnancement *période par période*. Dans l'ordonnancement global, tous les jobs à ordonnancer sur tout l'horizon de planification sont considérés en même temps ; tandis que dans la politique période par période, le problème d'ordonnancement est décomposé en T (nombre de périodes) problèmes d'ordonnancement, chaque problème étant indépendant des autres. La stratégie d'ordonnancement global conduit vers de meilleurs résultats.

Approche de Roux *et al.* [193]

Dans ce travail, un problème multi-niveaux et multi-sites est considéré. Une stratégie de résolution itérative similaire à celle utilisée dans [144] et [47] est appliquée, avec la différence qu'à chaque itération plusieurs problèmes d'ordonnancement sont résolus. En effet, comme il s'agit d'un réseau avec plusieurs sites de production, le problème d'ordonnancement peut être résolu de façon indépendante par site. À chaque itération, pour chaque plan de production généré, le problème d'ordonnancement est résolu avec une métaheuristique proposée dans [49].

Deux modèles mathématiques MIP pour le problème de planification avec séquence fixée et des contraintes de capacité détaillées par site de production sont proposés. Ils peuvent être vus comme des extensions du CLSP. Le premier modèle autorise la rupture de stock pour les produits finis. Par contre, afin d'assurer l'approvisionnement de la demande interne, les composants ne peuvent pas être en rupture de stock. Ainsi, des demandes externes sont considérées uniquement

pour les produits finis. Le deuxième modèle considère que tous les produits, y compris les composants à tous les niveaux, peuvent avoir des demandes externes. Au même titre, la rupture de stock est autorisée pour tous les produits. De nouvelles contraintes sont introduites afin de continuer à garantir la satisfaction de la demande interne.

Dans les deux modèles, les coûts et les temps de setup ne sont pas considérés. La fonction objectif consiste à minimiser la somme des coûts de fabrication, de stockage et de rupture des stocks.

La méthode a été testée sur 4 instances d'un problème d'ordonnancement dans un atelier de type job-shop flexible, avec 5 sites de production et 6 périodes. Les demandes externes ont été considérées uniquement sur les produits finis. Le problème de planification avec séquence fixée a été résolu au moyen de la librairie d'optimisation OSL d'IBM, et les problèmes d'ordonnancement par site ont été résolus en parallèle.

Modèle intégré de Dauzère-Pérès et Lasserre [48]

Les auteurs présentent une nouvelle version du modèle intégré proposé dans [144] et [47], permettant maintenant de résoudre des problèmes multi-niveaux et de tenir compte des délais d'obtention des produits. L'aspect multi-niveaux est modélisé en incluant les liens dans la nomenclature (*gozinto factors*) entre les produits et les tailles de lots des produits successeurs dans les contraintes d'équilibre de stocks.

Les auteurs décrivent l'intérêt d'intégrer les contraintes d'ordonnancement dans le problème de planification de la production, afin de garantir des solutions réalisables. Cependant, en tenant compte du fait que le modèle est très difficile à résoudre, ils recommandent l'utilisation de l'approche itérative initialement proposé dans [144] et [47] et étendue au cas multi-niveaux et multi-sites dans [193].

Approche de Ouerfelli *et al.* [175]

Un algorithme de résolution exacte, basé sur la méthode de décomposition de Benders [28], est présenté pour résoudre un problème mono-niveau. Le travail est inspiré de l'approche de Dauzère-Pérès et Lasserre [47], avec une différence de conception importante : l'ordonnancement est réalisé période par période. Le problème est décomposé en un problème principal qui fixe la séquence et un problème secondaire qui calcule le plan de production associé à la séquence précédemment fixée. Le problème principal est une version relâchée (éliminant les circuits) du problème d'ordonnancement.

La méthode de Benders est utilisée pour combiner de manière itérative la résolution du problème principal et du problème secondaire. Deux versions de la méthode de Benders (la méthode traditionnelle et la méthode avec des contraintes pour éliminer les circuits) sont comparées avec la résolution du modèle par le solveur commercial IBM ILOG CPLEX.

L'approche a été testée sur 7 instances sur un atelier de type job-shop avec 6 produits et 6 machines et 3 périodes de planification. Les solutions obtenues par IBM ILOG CPLEX sont meilleures dans tous les cas, et les écarts entre les bornes inférieure et supérieure des deux variantes de Benders sont très élevés (273% au minimum). Parmi les deux techniques de Benders, la méthode avec contraintes pour éliminer les circuits offre une meilleure performance. Il faut noter que seules des instances très petites, avec 3 périodes, sont résolues.

Première approche de Li et Ierapetritou [152]

Les auteurs résolvent un problème typique rencontré dans les industries de process. Afin de diminuer la difficulté de résolution du problème intégré, une structure d'horizon glissant est utilisée, afin de résoudre plusieurs sous-problèmes de planification avec capacité agrégée sur tout l'horizon de planification et plusieurs sous-problèmes d'ordonnancement (un par période), permettant ainsi de modifier les décisions tactiques pour obtenir des plans de production réalisables.

À chaque itération, les décisions sont mises à jour. Les tailles de lots obtenues avec l'ordonnancement détaillé remplacent les anciennes tailles de lots déterminées avec capacité agrégée, et les demandes non satisfaites à une période donnée sont reportées à la période suivante. L'algorithme de cette méthode est schématisé dans la Figure 2.7, où p_c, T, P_1 et P_2 correspondent à la période courante, au nombre total de périodes, au problème de planification et au problème d'ordonnancement, respectivement.

Les caractéristiques d'ordonnancement considérées dans le modèle sont :

- Non préemption,
- Capacité de la ressource (bornes inférieure et supérieure),
- Durées opératoires,
- Précédence entre les opérations,
- Équilibre de la matière,
- Capacité de stockage limitée,

Une contribution de ce travail par rapport aux méthodes traditionnelles d'horizon glissant, est que des contraintes de capacité, correspondant à des bornes sur la quantité de production de chaque produit (des inégalités valides), sont incluses dans le modèle de planification. Une

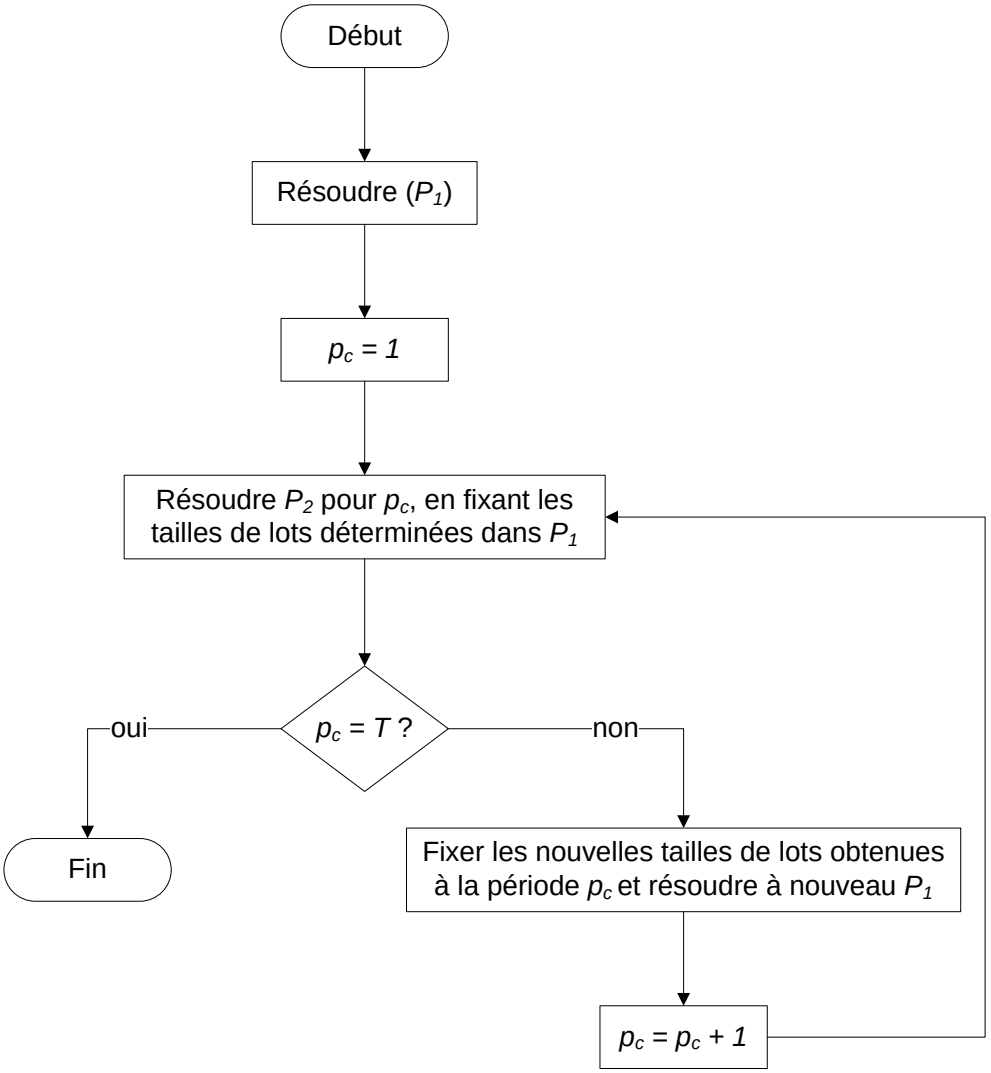

FIGURE 2.7 – Approche avec horizon glissant [152]

comparaison réalisée avec une modélisation sans contraintes de capacité montre que cette information permet d'améliorer de manière significative la qualité des résultats. L'amélioration est due principalement à une grande diminution des quantités en rupture de stock.

Les auteurs proposent aussi un algorithme de programmation paramétrique [183] pour obtenir une information plus précise sur la capacité. Cette technique permet d'obtenir des meilleurs résultats par rapport à la stratégie précédente (avec des contraintes de capacité simples). En considérant un système avec deux produits, ils déterminent la région géométrique de la capacité de production, en maximisant la taille de lot d'un produit et en faisant varier celle de l'autre, et vice versa. L'inconvénient est que lorsque le nombre de produits augmente, construire toutes les

régions de capacité devient très coûteux en termes d'effort de calcul. C'est pourquoi les auteurs proposent une heuristique permettant de décomposer le problème en plusieurs sous-problèmes contenant peu de produits.

L'avantage des procédures avec horizon glissant est que le modèle mathématique est réduit de façon importante, et par conséquent le temps de calcul aussi pour résoudre ce modèle. De plus, dans la pratique, cette stratégie peut permettre de tenir compte des incertitudes pouvant se présenter à court terme, et de recalculer les décisions prises pour le moyen terme. Un inconvénient est que l'écart entre les solutions proposées et les solutions optimales peut être grand.

Deuxième approche de Li et Ierapetritou [150]

Dans ce travail, un problème multi-produits et multi-opérations de l'industrie chimique est étudié. Le problème d'optimisation est modélisé à deux niveaux. Dans le niveau supérieur le problème de planification est formulé, et dans le niveau inférieur plusieurs sous-problèmes d'ordonnancement sont modélisés. Comme le modèle est très difficile à résoudre, une méthode de décomposition est proposée. L'objectif dans le modèle de planification est de minimiser la somme des coûts de stockage, de rupture et de production. Le modèle d'ordonnancement minimise le coût de production. Ce coût est composé d'une partie fixe et d'une partie variable qui dépend des tailles des lots. Les valeurs des variables de décision déterminées au niveau supérieur sont fixées comme paramètres dans le niveau inférieur. Les même contraintes d'ordonnancement que dans [152] sont considérées dans ce travail.

Pour éviter l'infaisabilité des sous-problèmes d'ordonnancement, le modèle d'ordonnancement est reformulé, en incluant un coût de pénalité dans la fonction objectif pour pénaliser les objectifs de production non atteints et des variables d'écart dans les contraintes fixant les objectifs de production, afin de représenter ceux qui ne sont pas atteints.

L'algorithme de résolution suit une méthode de décomposition qui itère entre la résolution d'un problème maître (modèle de planification sans contraintes d'ordonnancement), générant une borne inférieure, et la résolution de l'ensemble des sous-problèmes d'ordonnancement, qui permet d'obtenir une borne supérieure. À chaque itération, le coût de production est mis à jour. Une sous-estimation convexe de la fonction du coût de production est réalisée au moyen d'une relaxation Lagrangienne.

Deux cas d'études sont testés, mettant en évidence la complexité de résolution du problème intégré avec l'approche proposée. Dans le premier cas, l'écart relatif avec la solution de CPLEX est de 25%, et dans le deuxième, l'ecart relatif entre la borne inférieure et la borne supérieure

est de 8.5% après 28 heures de calcul.

Troisième approche de Li et Ierapetritou [151]

Les auteurs proposent une méthode de relaxation Lagrangienne augmentée, méthode qui a comme inconvénient la non-séparabilité du problème relâché. C'est pourquoi, les auteurs proposent une méthode d'optimisation à deux niveaux, permettant de décomposer ce problème. Dans le premier niveau, le problème relâché est résolu, en respectant seulement les variables de planification à travers un algorithme itératif. Dans le deuxième niveau, un ensemble de sous-problèmes d'ordonnancement est résolu, en fixant les variables de décision de planification déterminées dans le niveau supérieur. L'implémentation d'une stratégie d'approximation quadratique diagonale [195] est aussi mise en place et une comparaison des performances est réalisée.

L'inconvénient de l'approche à deux niveaux est qu'il est nécessaire d'optimiser un problème quadratique non différentiable à chaque itération, et donc l'effort de calcul devient important. D'autre part, à travers l'approche d'approximation quadratique diagonale, on résout une approximation du problème relâché et il n'est pas possible de garantir l'obtention de l'optimum global. Les comparaisons réalisées permettent de conclure que la méthode d'approximation est meilleure que la stratégie d'optimisation à deux niveaux.

Le modèle utilisé inclut toutes les contraintes de planification et d'ordonnancement dans une seule formulation, avec des variables de planification sur les deux types de contraintes (planification et ordonnancement) et des variables d'ordonnancement uniquement sur les contraintes d'ordonnancement. Pour coupler les deux types de contraintes, une reformulation est faite, en considérant des nouvelles variables servant à définir des contraintes couplantes. Les nouvelles contraintes sont relâchées, et comme expliqué précédemment, le problème est décomposé en un sous-problème de planification et plusieurs sous-problèmes d'ordonnancement. L'algorithme de résolution utilisé est illustré par la Figure 2.8, où λ et μ sont les multiplicateurs Lagrangiens, σ et le paramètre de pénalité, Inv et P sont les variables de tailles de lot et d'inventaire, PP et II sont des variables auxiliaires dupliquant Inv et P, respectivement, f est la valeur objectif du problème dual, g est la valeur de la fonction de consistance, T est le nombre de périodes et ϵ est l'écart de dualité. Les variables auxiliaires permettent de reformuler le problème à partir d'une décomposition.

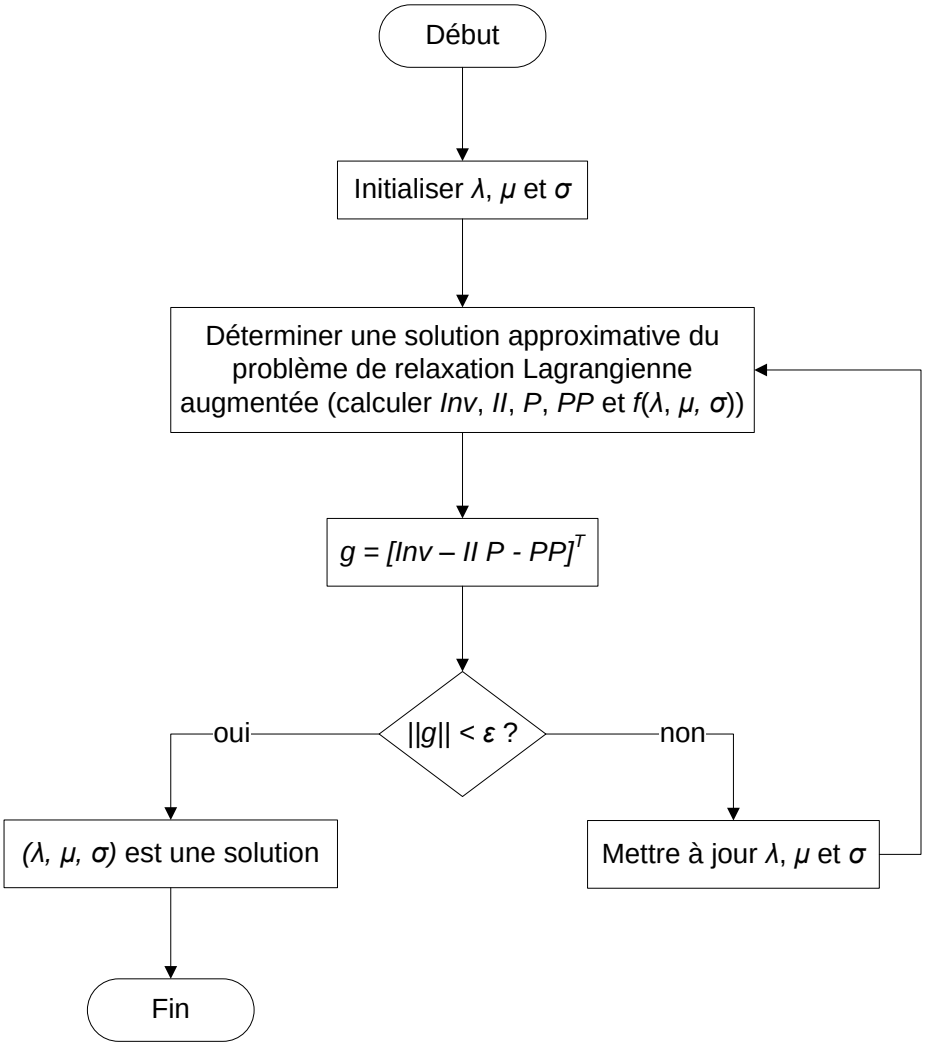

FIGURE 2.8 – Approche avec relaxation Lagrangienne augmentée [151]

2.5.2 Approches intégrées

Approche de Zhang et Yan [243]

Cette approche est basée sur un algorithme génétique hybride, et se focalise sur la résolution du problème de planification et d'ordonnancement dans un atelier de type job-shop. Les règles de priorité LPT (plus longue durée de fabrication) et SPT (plus courte durée de fabrication) pour générer la première séquence sont utilisées, le problème est formulé avec un modèle non-linéaire en variables mixtes et l'ordonnancement est réalisé période par période.

Peu d'instances sont résolues et des comparaisons avec des méthodes exactes n'ont pas été effectuées. Il est par conséquent difficile de conclure sur la performance de l'approche. Des comparaisons ont été réalisées uniquement avec une méthode hiérarchique, où l'on valide la pertinence d'intégrer des décisions.

Approche de Wolosewicz *et al.* [233, 234, 235]

Inspirée des travaux de [47] et [48], cette approche a été proposée dans le but de diminuer les temps de calcul et d'améliorer la qualité des résultats. La méthode itérative devient une méthode intégrée, où le problème d'ordonnancement n'est pas résolu à chaque itération, mais la séquence est modifiée à travers une métaheuristique (recherche taboue ou recuit simulé), permettant potentiellement d'éviter la convergence vers un optimum local et d'explorer de nouveaux espaces de recherche, afin de trouver une solution proche de l'optimum global. Comme dans les travaux précurseurs, l'ordonnancement est considéré sur tout l'horizon de planification.

Une heuristique Lagrangienne pour résoudre le problème de planification avec séquence fixée (voir [234] et [235]) et une recherche taboue pour guider l'évolution de la séquence sont mises en œuvre. Ainsi, les contraintes de capacité détaillées sont relâchées à travers la relaxation Lagrangienne, et l'algorithme dynamique de dimensionnement de lots proposé dans [230] est utilisé pour résoudre un ensemble de problèmes mono-produit sans contraintes de capacité. Pour réparer les contraintes de capacité, une heuristique de lissage de la production est couplée à l'heuristique Lagrangienne, permettant ainsi de calculer des bornes supérieures (plans de production réalisables). Cette heuristique modifie les tailles de lots en déplaçant des quantités de produit entre différentes périodes, avec le but d'éliminer la violation des contraintes de capacité. La difficulté majeure de l'heuristique de lissage est que, comme l'ordonnancement est réalisé sur tout l'horizon de planification, les déplacements de quantités de production réalisés entre deux périodes modifient potentiellement les dates de début et de fin de toutes les opérations, entre et après ces deux périodes et changent ainsi la capacité utilisée dans un très grand nombre de périodes. Néanmoins, comme montré dans [47], l'ordonnancement global est préférable à l'ordonnancement période par période, car la capacité est utilisée plus efficacement et de meilleures solutions sont trouvées.

Les deux versions de cette approche (avec recherche taboue et avec recuit simulé) et l'approche de Dauzère-Pérès et Lasserre [47] sont comparées, en résolvant plusieurs instances (en faisant varier la capacité, les coûts de setup, les demandes et les délais d'obtention) associées à différents ateliers de type job-shop et avec différents horizons de planification. La nouvelle approche avec recherche taboue donne globalement des meilleurs résultats.

La performance de l'approche a été aussi comparée avec la résolution exacte du modèle par le solveur commercial IBM ILOG CPLEX, et une fois de plus l'intérêt d'utiliser cette méthode est validée. Les solutions obtenues avec l'approche intégrée sont globalement meilleures que celles du solveur, surtout pour les grandes instances (avec 10 produits et 10 machines ou 20 produits et 5 machines et 10 ou 20 périodes), pour lesquelles IBM ILOG CPLEX n'obtient pas de solutions réalisables.

Le schéma sur la Figure 2.9 présente l'évolution de l'approche de Lasserre [144], jusqu'à l'approche de Wolosewicz *et al.* [233, 234, 235].

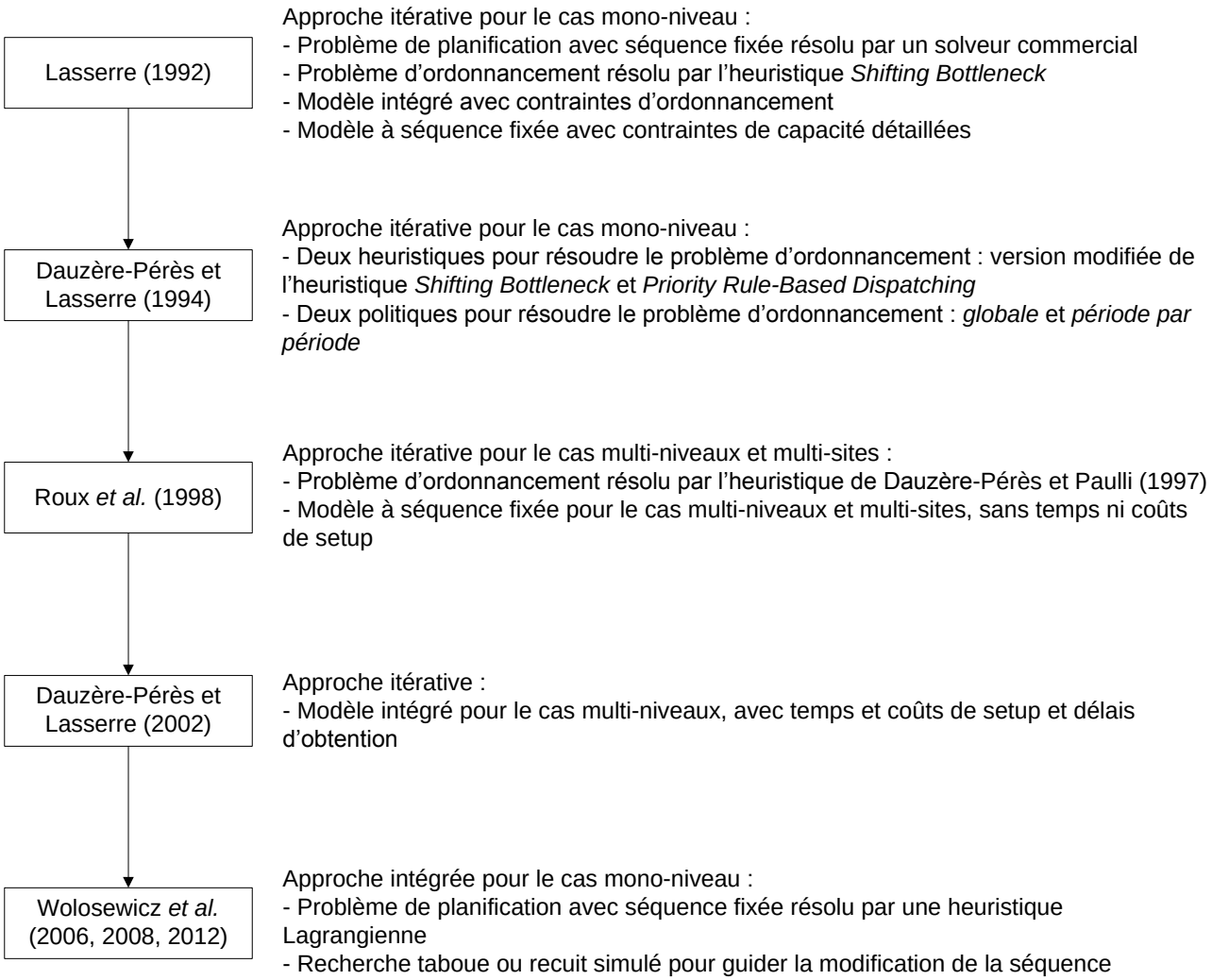

FIGURE 2.9 – Évolution de l'approche de Lasserre [144]

Approche de Shah et Ierapetritou [203]

Les auteurs étudient un problème mono-niveau et multi-sites avec coûts de transport, où plusieurs marchés peuvent être satisfaits par plusieurs sites de production, avec des coûts de transport différents. Une relaxation Lagrangienne augmentée avec une méthode d'approximation diagonale est implémentée pour résoudre le problème intégré. La non-satisfaction de la demande à une période donnée est autorisée dans le modèle, mais elle doit être satisfaite à la période suivante. Des coûts de rupture sont considérés à cet effet. L'objectif du problème intégré est de minimiser la somme des coûts de stockage, de rupture, de production et de transport. Les contraintes d'ordonnancement utilisées dans le modèle sont celles considérées dans [152] et [150].

Comme dans [151], les auteurs montrent que le modèle a une structure de bloc angulaire, où les variables de planification lient les contraintes de planification et d'ordonnancement. Pour décomposer le problème, ils proposent de transformer les variables compliquées (variables de planification) en contraintes compliquées, et de relâcher ces dernières. Dans la reformulation, les variables de planification sont dupliquées, de façon à utiliser des variables différentes pour la planification et l'ordonnancement, et des contraintes couplant ces deux types de variables sont incluses dans le modèle.

Une relaxation Lagrangienne augmentée est appliquée pour décomposer le problème. La nouvelle fonction objectif inclut des multiplicateurs Lagrangiens et un paramètre de pénalité quadratique. L'inconvénient qui apparaît à ce stade est la non-séparabilité des termes bilinéaires associés au paramètre de pénalité. Ce problème est contourné en appliquant une méthode d'approximation quadratique diagonale, qui permet de linéariser ces termes. Ainsi, une nouvelle reformulation est proposée, en décomposant la fonction objectif en deux sous-fonctions, une pour le problème de planification et une autre pour le problème d'ordonnancement. L'approche de résolution itère entre la résolution d'un problème de planification et la résolution de plusieurs sous-problèmes d'ordonnancement. Le modèle MIP du problème intégré original est transformé en un modèle quadratique pour le problème de planification et en des modèles quadratiques à nombres mixtes (MIQP) pour les sous-problèmes d'ordonnancement. L'algorithme converge vers une solution réalisable, mais l'optimalité n'est pas garantie parce que c'est une approximation du problème relâché qui est résolue.

La performance de l'approche a été comparée avec celle d'une méthode exacte utilisant le modèle intégré original et le solveur IBM ILOG CPLEX (version 12.2). Trois types de configurations ou systèmes ont été testés. Sur deux configurations, les résultats sont favorables au solveur standard pour les instances avec un nombre de périodes d'ordonnancement inférieur à

90 ; et pour l'autre configuration, IBM ILOG CPLEX est plus avantageux sur des instances avec un nombre de périodes inférieur à 45. Pour un nombre de périodes d'ordonnancement plus élevé, l'approche proposée donne de meilleures solutions.

2.6 Conclusion

Nous avons présenté une perspective générale sur la façon dont le lien entre la planification et l'ordonnancement de la production et dans la chaîne logistique est étudié dans la littérature, afin de réduire l'incohérence des décisions hiérarchiques proposées par les approches classiques du type MRP-II, et de déterminer des plans de production tactiques réalisables au niveau opérationnel.

Dans un premier temps, nous avons posé les bases de la complexité, la modélisation et la résolution des problèmes d'optimisation liées aux étapes de planification de la chaîne logistique, et plus précisément aux problèmes de dimensionnement de lots et d'ordonnancement. Ensuite, nous avons présenté une vue d'ensemble des problèmes de dimensionnement de lots multi-produits avec contraintes de capacité et demande dynamique, en utilisant la classification classique selon la longueur des périodes et le nombre de niveaux de nomenclature. Ainsi, nous avons vu que les problèmes à courtes périodes (DLSP, CSLP et PLSP) restreignent l'utilisation de la capacité et considèrent un nombre important de variables discrètes, ce qui rend difficile la résolution de ces problèmes. En raison des hypothèses des modèles, seuls des systèmes de production simples (ateliers à une machine ou à machines parallèles) peuvent être considérés. Parmi les modèles à longues périodes, le GLSP a moins de limitations sur la modélisation de la capacité, mais la difficulté de résolution augmente fortement ; tandis que le CLSP modélise la capacité de manière globale, mais il n'est pas adapté à la résolution des systèmes de production complexes, et la prise en compte de contraintes d'ordonnancement entraîne une difficulté supplémentaire très importante. Pour ce qui relève de la prise en compte de la nomenclature, nous avons décrit les travaux les plus représentatifs sur les systèmes mono-niveau et multi-niveaux à une et plusieurs ressources. Peu de travaux ont étudié les problèmes multi-niveaux, et encore moins ceux à plusieurs ressources, malgré leur pertinence dans de nombreux environnements industriels réels. La raison est la complexité des problèmes, notamment avec l'utilisation des modèles mathématiques dont nous avons discuté.

En ce qui concerne les problèmes d'ordonnancement (sans contraintes ni objectifs de planification tactique), nous avons brièvement décrit les principaux groupes de problèmes traités dans la littérature, et nous avons donné des généralités sur la complexité et les méthodes de résolution.

Une attention spéciale a été accordée aux problèmes de planification intégrant des contraintes d'ordonnancement, dans le but de déterminer des solutions réalisables et proches de l'optimum, pour des systèmes impliquant le partage de ressources entre différents produits et nécessitant le traitement de plusieurs opérations. À cet effet, nous avons présenté les approches itératives et les approches intégrées existantes dans la littérature, ainsi que leurs limitations et avantages. Parmi ces travaux, nous retenons l'approche de Wolosewicz *et al.* [233, 234, 235], pour laquelle nous proposons des améliorations significatives dans le Chapitre 3 et une extension aux problèmes multi-niveaux dans le Chapitre 4.