NNT : 2012 EMSE 0672

THÈSE

présentée par

Sergey KOVALEV

pour obtenir le grade de
Docteur de l'École Nationale Supérieure des Mines de Saint-Étienne

Spécialité : Génie Industriel

PROBLÈMES COMBINATOIRES EN CONFIGURATION DES LIGNES DE FABRICATION : ANALYSE DE COMPLEXITÉ ET OPTIMISATION

soutenue à Saint-Étienne, le 23 novembre 2012

Membres du jury

Rapporteurs :	Farouk YALAOUI	Professeur, Université de technologie de Troyes
	Christian ARTIGUES	Directeur de recherche, LAAS-CNRS, Toulouse
Examinateur(s) :	Abdelhakim ARTIBA	Professeur, Université de Valenciennes
	Ammar OULAMARA	Professeur, Université de Lorraine, Metz
	Frank WERNER	Professeur, Otto-von-Guericke-University, Magdeburg
	François VANDERBECK	Professeur, Université Bordeaux 1, Talence
Directeur(s) de thèse :	Alexandre DOLGUI	Professeur, École des Mines de Saint-Étienne
Co-encadrant de thèse :	Xavier DELORME	Maître-assistant, École des Mines de Saint-Étienne

Spécialités doctorales :	Responsables :
SCIENCES ET GENIE DES MATERIAUX	K. Wolski Directeur de recherche
MECANIQUE ET INGENIERIE	S. Drapier, professeur
GENIE DES PROCEDES	F. Gruy, Maître de recherche
SCIENCES DE LA TERRE	B. Guy, Directeur de recherche
SCIENCES ET GENIE DE L'ENVIRONNEMENT	D. Graillot, Directeur de recherche
MATHEMATIQUES APPLIQUEES	O. Roustant, Maître-assistant
INFORMATIQUE	O. Boissier, Professeur
IMAGE, VISION, SIGNAL	JC. Pinoli, Professeur
GENIE INDUSTRIEL	A. Dolgui, Professeur
MICROELECTRONIQUE	

EMSE : Enseignants-chercheurs et chercheurs autorisés à diriger des thèses de doctorat (titulaires d'un doctorat d'État ou d'une HDR)

AVRIL	Stéphane	PR2	Mécanique et ingénierie	CIS
BATTON-HUBERT	Mireille	PR2	Sciences et génie de l'environnement	FAYOL
BENABEN	Patrick	PR1	Sciences et génie des matériaux	CMP
BERNACHE-ASSOLLANT	Didier	PR0	Génie des Procédés	CIS
BIGOT	Jean Pierre	MR(DR2)	Génie des Procédés	SPIN
BILAL	Essaid	DR	Sciences de la Terre	SPIN
BOISSIER	Olivier	PR1	Informatique	FAYOL
BORBELY	Andras	MR(DR2)		SMS
BOUCHER	Xavier	PR2	Génie Industriel	FAYOL
BRODHAG	Christian	DR	Sciences et génie de l'environnement	FAYOL
BURLAT	Patrick	PR2	Génie Industriel	FAYOL
COLLOT	Philippe	PR0	Microélectronique	CMP
COURNIL	Michel	PR0	Génie des Procédés	DIR
DARRIEULAT	Michel	IGM	Sciences et génie des matériaux	SMS
DAUZERE-PERES	Stéphane	PR1	Génie Industriel	CMP
DEBAYLE	Johan	CR	Image Vision Signal	CIS
DELAFOSSE	David	PR1	Sciences et génie des matériaux	SMS
DESRAYAUD	Christophe	PR2	Mécanique et ingénierie	SMS
DOLGUI	Alexandre	PR0	Génie Industriel	FAYOL
DRAPIER	Sylvain	PR1	Mécanique et ingénierie	SMS
FEILLET	Dominique	PR2	Génie Industriel	CMP
FOREST	Bernard	PR1	Sciences et génie des matériaux	CIS
FORMISYN	Pascal	PR0	Sciences et génie de l'environnement	DIR
FRACZKIEWICZ	Anna	DR	Sciences et génie des matériaux	SMS
GARCIA	Daniel	MR(DR2)	Génie des Procédés	SPIN
GIRARDOT	Jean-jacques	MR(DR2)	Informatique	FAYOL
GOEURIOT	Dominique	DR	Sciences et génie des matériaux	SMS
GRAILLOT	Didier	DR	Sciences et génie de l'environnement	SPIN
GROSSEAU	Philippe	DR	Génie des Procédés	SPIN
GRUY	Frédéric	PR1	Génie des Procédés	SPIN
GUY	Bernard	DR	Sciences de la Terre	SPIN
GUYONNET	René	DR	Génie des Procédés	SPIN
HAN	Woo-Suck	CR		SMS
HERRI	Jean Michel	PR1	Génie des Procédés	SPIN
INAL	Karim	PR2	Microélectronique	CMP
KLOCKER	Helmut	DR	Sciences et génie des matériaux	SMS
LAFOREST	Valérie	MR(DR2)	Sciences et génie de l'environnement	FAYOL
LERICHE	Rodolphe	CR	Mécanique et ingénierie	FAYOL
LI	Jean Michel		Microélectronique	CMP
MALLIARAS	Georges	PR1	Microélectronique	CMP
MOLIMARD	Jérôme	PR2	Mécanique et ingénierie	CIS
MONTHEILLET	Franck	DR	Sciences et génie des matériaux	SMS
PERIER-CAMBY	Laurent	PR2	Génie des Procédés	DFG
PIJOLAT	Christophe	PR0	Génie des Procédés	SPIN
PIJOLAT	Michèle	PR1	Génie des Procédés	SPIN
PINOLI	Jean Charles	PR0	Image Vision Signal	CIS
ROUSTANT	Olivier	MA(MDC)		FAYOL
STOLARZ	Jacques	CR	Sciences et génie des matériaux	SMS
SZAFNICKI	Konrad	MR(DR2)	Sciences et génie de l'environnement	CMP
TRIA	Assia		Microélectronique	CMP
VALDIVIESO	François	MA(MDC)	Sciences et génie des matériaux	SMS
VIRICELLE	Jean Paul	MR(DR2)	Génie des Procédés	SPIN
WOLSKI	Krzystof	DR	Sciences et génie des matériaux	SMS
XIE	Xiaolan	PR1	Informatique	CIS

ENISE : Enseignants-chercheurs et chercheurs autorisés à diriger des thèses de doctorat (titulaires d'un doctorat d'État ou d'une HDR)

FORTUNIER	Roland	PR	Sciences et Génie des matériaux	ENISE
BERGHEAU	Jean-Michel	PU	Mécanique et Ingénierie	ENISE
DUBUJET	Philippe	PU	Mécanique et Ingénierie	ENISE
LYONNET	Patrick	PU	Mécanique et Ingénierie	ENISE
SMUROV	Igor	PU	Mécanique et Ingénierie	ENISE
ZAHOUANI	Hassan	PU	Mécanique et Ingénierie	ENISE
BERTRAND	Philippe	MCF	Génie des procédés	ENISE
HAMDI	Hédi	MCF	Mécanique et Ingénierie	ENISE
KERMOUCHE	Guillaume	MCF	Mécanique et Ingénierie	ENISE
RECH	Joël	MCF	Mécanique et Ingénierie	ENISE
TOSCANO	Rosario	MCF	Mécanique et Ingénierie	ENISE
GUSSAROV Andrey	Andrey	Enseignant contractuel	Génie des procédés	ENISE

PR 0	Professeur classe exceptionnelle
PR 1	Professeur 1ère classe
PR 2	Professeur 2ème classe
PU	Professeur des Universités
MA (MDC)	Maître assistant
DR	Directeur de recherche

Ing.	Ingénieur
MCF	Maître de conférences
MR (DR2)	Maître de recherche
CR	Chargé de recherche
EC	Enseignant-chercheur
IGM	Ingénieur général des mines

SMS	Sciences des Matériaux et des Structures
SPIN	Sciences des Processus Industriels et Naturels
FAYOL	Institut Henri Fayol
CMP	Centre de Microélectronique de Provence
CIS	Centre Ingénierie et Santé

Mise à jour : 04/09/2012

Remerciements

Je voudrais exprimer ma reconnaissance aux personnes sans lesquelles cette thèse n'aurait pas pu apparaître.

Tout d'abord, je voudrais remercier Monsieur Alexandre Dolgui, mon directeur de thèse et Professeur à l'École des Mines de Saint-Étienne, pour m'avoir fait confiance et pour l'organisation de très bonnes conditions de travail. Sa compétence et son enthousiasme ont joué un rôle très important dans la réussite de cette thèse.

J'exprime ma profonde gratitude à Monsieur Xavier Delorme, mon co-encadrant de thèse et Maître-Assistant à l'École des Mines de Saint-Étienne. Ses conseils précieux et sa vision sur notre recherche m'ont aidé non seulement à améliorer la thèse mais ont aussi donné la possibilité de voir différents aspects des problèmes traités sous un autre angle.

Je suis très reconnaissant à tous ceux avec qui j'ai collaboré au cours des trois dernières années. Je cite Pavel Borisovski, Ammar Oulamara, Erwin Pesch et Jenny Nossack. Ils ont beaucoup contribué au succès de notre recherche. Leur professionalisme et bienveillance m'ont énormément impressionné.

Je tiens à remercier les membres du jury qui m'ont fait l'honneur d'accepter d'évaluer mon travail. Je remercie Farouk Yalaoui, Professeur à l'Université de Technologie de Troyes, et Christian Artigues, Directeur de Recherche au LAAS/CNRS, de m'avoir fait l'honneur et le plaisir de rapporter ma thèse. Je remercie également Abdelhakim Artiba, Professeur à l'Université de Valenciennes, Ammar Oulamara, Professeur à l'Université de Lorraine, Frank Werner, Professeur à l'Université Otto-von-Guericke et François Vanderbeck, Professeur à l'Université Bordeaux 1, pour avoir accepté d'examiner ce mémoire.

J'adresse un grand merci à toute l'équipe DEMO de l'Institut Fayol pour leur soutien et leur aide que j'apprécie beaucoup.

Je suis particulièrement reconnaissant à mes nombreux amis de presque tous les continents avec qui j'ai partagé l'une des meilleures parties de ma vie. Grâce à eux, mon séjour à Saint-Étienne a été très agréable.

Enfin, je remercie ma famille qui me soutient toujours et donne la force de surmonter les épreuves.

Table des matières

Liste des tableaux

Introduction générale

Dans l'industrie contemporaine, les systèmes de production jouent un rôle très important. La population mondiale augmente rapidement, ce qui suscite une demande croissante pour des produits et des services. Afin de satisfaire cette demande et rivaliser avec succès avec les concurrents, chaque entreprise doit soigneusement concevoir son système de production. Il faut créer un système qui sera à la fois efficace du point de vue de la qualité des produits fabriqués et du point de vue du capital dépensé. L'un des systèmes de production les plus utilisés dans l'industrie est la ligne de fabrication. Elle se compose de postes de travail (stations) alignés de manière séquentielle. Les décisions liées à la conception des lignes de fabrication sont de grande importance parce que le coût de la mise en place d'une ligne de fabrication se chiffre en millions d'euros.

Dans ce mémoire, nous nous intéressons particulièrement à l'étape de la configuration des lignes de fabrication. Cette étape suit l'analyse des produits à fabriquer et la planification du processus de fabrication. La configuration des lignes consiste à choisir les ressources de production et à les affecter à des stations de sorte que les objectifs définis soient atteints. Ces objectifs peuvent être représentés, par exemple, par la minimisation du coût total ou la maximisation de la productivité de la ligne. Les problèmes survenant au stade de la configuration des lignes de fabrication sont de nature combinatoire. Le concepteur de la ligne cherche la(les) solution(s) optimale(s) parmi les autres solutions admissibles. C'est une tâche difficile en tenant compte d'une explosion combinatoire des solutions possibles. Ainsi, le choix d'une méthode de résolution dans ces conditions est décisif. La méthode appliquée doit être efficace par rapport à la qualité des solutions obtenues et au temps de calcul.

Dans cette thèse, nous considérons deux types de problèmes de configuration des lignes de fabrication. Ces problèmes ont été notamment observé dans le domaine de l'usinage. Pour au-

tant, ces hypothèses et caractéristiques peuvent s'appliquer à des problèmes provenant d'autres domaines. Nous poursuivons deux buts : analyser la complexité de ces problèmes et trouver une méthode d'optimisation adaptée. Dans ce qui suit, nous présentons la structure de la thèse qui se compose de cinq chapitres.

Dans le chapitre 1, nous donnons une définition des lignes de fabrication et proposons une classification basée sur les critères de variabilité et de flux de produits. Nous donnons également quelques définitions importantes concernant la notion de temps de cycle. Ensuite, nous positionnons l'étape de configuration dans le processus de conception de lignes de fabrication et soulignons son importance.

Dans le chapitre 2 nous présentons un état de l'art sur les problèmes de configuration de lignes de fabrication. Nous notons que la configuration de lignes contient non seulement l'affectation des ressources aux stations (équilibrage) mais aussi leur sélection parmi les ressources disponibles. Nous donnons une nouvelle classification des problèmes d'équilibrage qui permet de distinguer clairement les lignes selon leur variabilité de production et de décrire en détail les caractéristiques de temps et de coût pour chaque type de ligne. En outre, cette classification permet de bien définir les problèmes abordés dans cette thèse. Dans le cadre de la classification proposée, un aperçu des problèmes considérés dans la littérature correspondant aux éléments de la classification est également donné. Puis, nous mentionnons les méthodes de résolution les plus répandues.

Le problème étudié dans le chapitre 3 consiste en même temps à sélectionner un sous-ensemble des ressources à partir d'un ensemble des ressources disponibles et à les affecter à un nombre de stations à déterminer de sorte que le coût total soit minimisé. La ligne est cadencée et elle est dédiée à la production d'un seul type de produit. Les stations sont alignées l'une après l'autre et toutes les pièces passent par toutes les stations dans le même ordre. Nous considérons ici le cas de lignes de fabrication avec des têtes d'usinage multi-broches. Les ressources correspondent à des modules de traitement (têtes multibroches que nous appellerons blocs) qui exécutent simultanément les opérations qui leur sont affectées. Les blocs affectées à la même station traitent un produit brut en parallèle. C'est à dire, les blocs sont simultanément activés. Un tel environnement de production augmente considérablement la productivité de la ligne. La

méthode de résolution proposée se base sur le concept d'une station localement admissible qui permet de réduire le problème étudié à un problème de partition d'ensemble.

Les chapitres 4 et 5 sont consacrés à deux problèmes de minimisation des coûts de changements de séries pour des lignes multi-produits flexibles ou reconfigurables. Ces deux problèmes partagent la plupart des mêmes hypothèses, quoiqu'il y ait des différences importantes dans leur formulation. Les hypothèses communes de ces deux problèmes sont les suivantes. La ligne est multi-produit, c'est-à-dire, elle doit être configurée pour la fabrication de produits de plusieurs types. La ligne est cadencée et il y a un cheminement de production unique pour tous les types de produits. Un ensemble d'opérations spécifiques est associé à chaque type de produits. Un changement de séries s'effectue sur une station si au moins une opération d'un certain type de produit est affectée à cette station. Chaque changement de séries implique des coûts complémentaires. Des contraintes de précédence sont données sur l'ensemble des opérations. Les problèmes ont deux critères considérés lexicographiquement : la minimisation du nombre des stations et la minimisation du coût total de changements de séries.

Les problèmes des chapitres 4 et 5 se distinguent par les hypothèses suivantes. Le problème dans le chapitre 4 suppose l'exécution parallèle des opérations sur les stations (comme dans le chapitre 3). Il y a des contraintes d'exclusion imposées sur un ensemble des opérations. Dans le chapitre 5 nous examinons un problème pour le cas avec l'exécution séquentielle des opérations sur les stations. Les contraintes d'exclusion ne sont pas présentes. Par contre, nous considérons les temps opératoires et les temps de cycle. Les produits de tous les types sont ordonnés selon le principe de la technologie de groupe. Ce principe implique que les produits d'un certain type entrent tous dans la ligne l'un après l'autre et qu'il n'y a pas de produits des autres types entre eux. Il y a une borne supérieure à respecter sur le temps de cycle du produit de chaque type.

Les procédures de résolution proposées dans les chapitres 4 et 5 se basent sur la même approche. D'abord, nous appliquons des heuristiques gloutonnes pour trouver des solutions approchées. Ensuite, nous modélisons le problème de la recherche du nombre de stations minimal comme un problème de programmation linéaire en nombres entiers et le résolvons à l'aide du solveur Cplex. En choisissant parmi les solutions obtenues celle qui a un nombre minimal de stations, nous construisons un modèle de programmation linéaire en nombres entiers pour le

problème de la recherche du coût total de changements de séries. Nous appliquons de nouveau le solveur pour trouver une solution optimale. En outre, nous établissons la complexité de calcul pour certains cas particuliers des problèmes considérés.

Finalement, nous terminons cette thèse par la conclusion générale où nous faisons le bilan des travaux effectués ainsi qu'une synthèse des principaux résultats obtenus. La discussion sur les perspectives de notre recherche y est également présentée.

Chapitre 1

Conception de lignes de fabrication

1.1 Systèmes de production

L'ensemble des moyens et des ressources utilisés pour fabriquer des produits finis à partir de produits bruts forme un **système de production**. Notons qu'un produit fini ne signifie pas forcément un produit qui est prêt pour l'utilisation par des consommateurs finaux. Il y a des producteurs qui fabriquent des composants pour d'autres producteurs. À leur tour, ceux-ci reçoivent (souvent via les centres logistiques) les composants comme les produits bruts pour fabriquer leur propre produit fini ayant une plus grande valeur ajoutée. Par exemple, dans la construction automobile, les producteurs de boîtes de vitesses transmettent leurs produits finis au producteur final qui assemble les produits prêts à vendre aux consommateurs, c'est-à-dire les voitures. Les boîtes de vitesses ne servent à rien sans voiture. Cependant, pour un producteur de boîtes de vitesses ses produits représentent des produits finis. Les producteurs peuvent être associés à deux types d'interactions économiques : B2B (Business to Business) - entreprise aux entreprises et B2C (Business to Customer) - entreprise aux particuliers. La différenciation se fait selon les clients ciblés. L'ensemble de ces producteurs et leurs interactions constitue une **chaîne d'approvisionnement**, connue dans la littérature anglo-saxonne sous le terme *supply chain*. Afin d'obtenir des produits finis il faut effectuer un ensemble d'opérations ordonné sur des produits bruts. Ces opérations sont exécutées par des ressources de production : soit par des opérateurs humains, soit par des machines automatiques. Il est évident que, s'agissant de production industrielle, même si des opérateurs participent au processus de production, de

toute façon, ils emploient certains outils ou machines. De même, un processus de production entièrement automatisé doit être supervisé par des spécialistes. Les ressources de production sont habituellement affectées à des postes de travail, plus souvent appelés des **stations**. Chaque station est caractérisée par des opérations qui lui sont affectées et qu'y sont exécutées de façon répétitive. Les produits bruts sont déplacés entre des stations où ils subissent des opérations. On obtient un produit fini lorsque la dernière opération sur la dernière station est finie. Un cheminement de production signifie une séquence de stations prédefinie pour un produit.

On peut distinguer trois types principaux de système de production, à savoir : les systèmes d'ateliers à cheminement multiple (*job-shop*), les systèmes d'ateliers à cheminement unique (*flow-line*) et les systèmes hybrides :

- Un **système d'ateliers à cheminement multiple** implique des cheminements de production différents pour chaque type de produit. La production est donc discontinue et flexible. Ce type de système de production est mis en œuvre pour la fabrication de produits dont les caractéristiques sont hétérogènes et pour lesquels la demande est petite. Compte tenu de l'utilisation de mêmes ressources pour la fabrication d'un ensemble de produits qui diffèrent parfois fortement les uns des autres, il est nécessaire d'ordonnancer les tâches pour utiliser les ressources au maximum et réduire le temps total de production. Vu la faible quantité et la très haute diversité de produits fabriqués, le niveau d'automatisation de ce type de système reste faible. Les ateliers flexibles (FMS - flexible manufacturing systems) qui devaient pallier à ce défaut coûtent très cher. Ils sont relativement complexes et demandent une certaine organisation du flux pour automatiser le transport des produits.
- Un **système d'ateliers à cheminement unique** signifie une séquence linéaire de stations auxquels des opérations sont affectées. Toutes les unités de produits de tous les types (s'il y a plusieurs types) ont un seul cheminement de production. Ainsi, la production a une haute cadence mais avec un degré de flexibilité plus faible que celui d'un système d'ateliers à cheminement multiple. Le flux de produits dans ce système suit la même direction. C'est-à-dire, l'entrée et la sortie de système sont toujours les mêmes. Ce système permet d'obtenir une productivité élevée et de satisfaire une forte demande. Le niveau d'automatisation est souvent élevé.
- Un **système de production hybride** réunit à la fois des caractéristiques d'ateliers à

cheminement multiple et unique. Cette symbiose est par exemple possible grâce aux **stations parallèles** qui permettent d'avoir plusieurs cheminements de production en gardant la même direction du flux de produits.

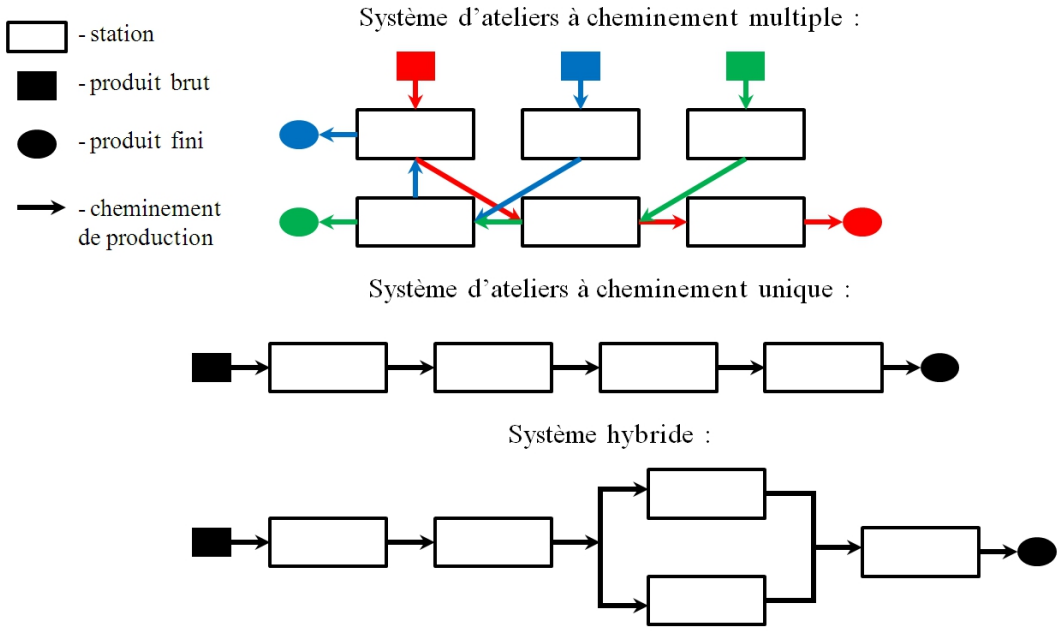


FIGURE 1.1 – Schémas illustratifs des types de systèmes de production

La Figure 1.1 présente les exemples schématiques de ces différents types de systèmes de production. La gestion d'un système d'ateliers à cheminement multiple est beaucoup plus complexe que celle d'un système d'ateliers à cheminement unique parce qu'elle nécessite d'ordonner la fabrication de plusieurs types de produits et de déterminer les tailles de lots [Garey et al., 1976, Petrovic et al., 2005]. Par contre, son avantage se trouve dans la capacité de fabriquer des produits dont les caractéristiques sont très différentes. Les deux premiers types de systèmes de production se distinguent entre eux par leurs niveaux de flexibilité et leurs taux de production. Le taux de procution d'un système à cheminement unique est plus élevé, tandis que la flexibilité est plus faible par rapport à un système d'atelier à cheminement multiple. Le choix du système à mettre en œuvre dépend de la nature de la production. Le système d'ateliers à cheminement multiple est essentiellement utilisé dans la production sur commande et la production en petites séries, tandis que le système d'ateliers à cheminement unique trouve son application principalement dans la production de masse. Ces trois types de systèmes de production sont décrits en détail dans [Tanaev et al., 1994, Pinedo and Chao, 1999].

1.2 Lignes de fabrication : définition et classification

Une **ligne de fabrication** est un système de production à cheminement unique ou hybride qui se compose de stations alignées de manière séquentielle.

Depuis leur apparition dans l'usine de Ford Motor Company, les lignes de fabrication sont devenues le moyen principal de production de masse. Les produits pour lesquels la demande est grande sont fabriqués essentiellement dans des usines équipées de lignes de fabrication. Les lignes de fabrication exploitent des avantages, tels que la division du travail, leur cadence de production, et les économies d'échelle, ce qui leur permet d'atteindre un haut niveau de productivité et de maintenir les coûts unitaires des produits à un bas niveau.

Historiquement les premières lignes de fabrication étaient essentiellement composées de postes de travail manuels situés le long d'un tapis roulant. Au fur et à mesure, afin de satisfaire à une augmentation de la productivité et de la qualité de produits, les machines automatiques ont été installées sur ces lignes. De plus, l'introduction de machines automatiques a permis de réduire le risque d'accident du travail sur les stations où la presence d'un opérateur humain n'est pas nécessaire tandis que le danger pour la santé est élevé.

Hormis leur architecture et leur degré d'automatisation plus ou moins grand, une classification des lignes de fabrication peut aussi être effectuée en considérant différents critères. Nous évoquons deux critères qui aident à décrire les types de lignes étudiées dans les chapitres 3, 4 et 5. Ce sont les critères de **variabilité** et de **flux de produits**.

La *variabilité* d'une ligne indique la fréquence de changements de types de produits fabriquées.

La notion de *flux de produits* sous-entend la caractéristique du mouvement de produits bruts sur une ligne de fabrication.

Nous pouvons distinguer trois types de lignes de fabrication en tenant compte du critère de **variabilité** :

1. Les **lignes de fabrication dédiées**. Ce sont les lignes conçues pour la fabrication d'un seul type de produit en grande série. Elles ont le plus grand volume de production parmi tous les systèmes de production. Cela permet d'avoir un coût réduit par unité de produit. En plus, vu que les ressources installées le long d'une ligne sont stables, la qualité de

produits se maintient sur un même niveau. Donc, une fois que les facteurs de production sont bien établis, une ligne dédiée permet d'obtenir un niveau de qualité de produits stable et élevé pendant une longue période de temps. Lorsque la demande est stable et importante, l'utilisation de lignes de fabrication dédiées rapporte ainsi un grand bénéfice. Par contre, la conception d'une telle ligne exige un examen méticuleux parce que son temps de fonctionnement peut s'étendre sur plusieurs années voire dizaines d'années. Par exemple, la fabrication des carrosseries du type W124 de Mercedes a duré 12 ans, de 1984 jusqu'à 1995 [w12, 2012]. Pendant ce temps la séquence des stations et l'ensemble des ressources qui leurs sont attribuées restaient les mêmes. L'autre défaut de ce type de lignes de fabrication vient des difficultés engendrées par des changements de spécification de produit. Après l'installation d'une telle ligne une reconfiguration devient complexe et très coûteuse.

2. Les **lignes de fabrication flexibles**. Au fil du temps, les exigences relatives aux produits ont sensiblement augmenté. La compétition dans les différentes branches de l'industrie et la demande croissante des consommateurs ont suscité l'apparition d'un nouveau concept de production, à savoir, la *personnalisation de masse*. Cette stratégie de production avancée signifie d'une part la possibilité de créer des produits individualisés selon le choix des consommateurs et d'autre part le maintien d'un relativement haut niveau du taux de production. L'un des exemples les plus réussis d'application de la stratégie de personnalisation de masse est la production d'ordinateurs par Dell. Des consommateurs peuvent commander des ordinateurs personnalisés selon leurs préférences sur le site Internet et puis, après que les commandes sont confirmées, les ordinateurs sont assemblés dans les usines de Dell. La compagnie utilise une règle appelée « fabrication sur commande » [Pollard et al., 2008]. Les autres compagnies fabriquant des ordinateurs se sont appropriées cette expérience avantageuse, par exemple, Sony qui est devenu le fournisseur d'ordinateurs les plus personnalisés. Un autre exemple d'une telle stratégie vient de l'industrie automobile avec le constructeur allemand BMW, qui propose un catalogue d'options pouvant théoriquement engendrer 10^{32} modèles différents [Meyr, 2004].

   Ce type de stratégie est mis en œuvre à l'aide des lignes de fabrication flexibles. Ce sont des lignes utilisées pour la fabrication de plusieurs types de produits en exploitant

la même séquence de stations. Pourtant, ces produits doivent avoir des caractéristiques similaires en termes de dimensions, formes géométriques et matériaux composants. La notion de *famille de produits* est souvent utilisée. En général, les ressources de production correspondent à des machines automatiques. La séquence d'opérations (à ne pas confondre avec la séquence de stations !) ainsi que les opérations exécutées en tant que telles différent en fonction du type de produits. Si, par exemple, un produit d'un certain type entre dans une ligne flexible, seules les ressources liées à la fabrication des produits de ce type seront activées, tandis que les ressources qui ne participent pas à la fabrication des produits de ce types resteront inactives. Certaines stations peuvent ainsi être inactives pendant la fabrication d'une unité de produit d'un type spécifique.

Les avantages des lignes flexibles sont évidents : la variabilité de produits fabriqués permet d'augmenter la compétitivité des entreprises et de cibler les différentes niches du marché. La possibilité de choisir un type de produit en effectuant une commande attire les consommateurs. Pourtant, il y a aussi des inconvénients. Le coût d'installation d'une telle ligne devient onéreux en raison d'utilisation de machines de type CNC (Computer Numeric Control). Ces machines sont souvent développées pour une famille de produits présupposée lorsque les exigences des produits finis et respectivement du processus de production ne sont pas connues avec précision [Delorme et al., 2009]. Cela pousse les constructeurs de machines CNC à inclure le plus de fonctionnalités possibles, ce qui augmente leur coût. La fiabilité des lignes flexibles est aussi plus basse par rapport aux lignes dédiées à cause de leur complexité.

3. Les **lignes de fabrication reconfigurables**. Une *reconfiguration* de ligne signifie un processus de changement de l'architecture de ligne et/ou de l'affectation des ressources afin de commencer la production de nouveaux types de produits avec une ancienne ligne de fabrication. C'est souvent un processus de longue durée qui exige beaucoup de dépenses. L'apparition de lignes capables de changer vite leur configuration physique a été suscitée par la tendance à la réduction du cycle de vie des produits et la demande croissante de personnalisation. Les lignes reconfigurables ont pour but de répondre plus vite et de manière moins coûteuse aux fluctuations du marché, en offrant à la fois le volume et la variabilité de production désirés [Koren et al., 1999]. La reconfigurabilité de telles

lignes est engendrée par la division des ressources de production en modules qui peuvent être facilement interchangés, intégrés ou enlevés. Les lignes reconfigurables visent à la fabrication d'une famille limitée de produits. La variabilité n'est prévue que pour ces produits. Donc, le coût d'investissement est plus bas par rapport aux lignes flexibles. En plus, il y a un avantage lié à la capacité des lignes reconfigurables. Dès qu'une ligne est (re)configurée, son taux de production peut s'élever à un niveau désiré, même jusqu'au niveau de lignes dédiées, si la demande de produit s'avère suffisamment haute. Pourtant, le coût par unité de produit reste plus haut par rapport à des lignes dédiées en raison de la complexité plus élevée des ressources (modules).

Il y a un phénomène très important qui est inhérent aux lignes flexibles et reconfigurables, à savoir le changement de séries. Un **changement de séries** signifie l'ensemble des démarches nécessaires pour préparer une station au traitement d'un produit brut d'un certain type. Ces démarches incluent : le réglage des machines, le chargement d'une pièce de produit, le positionnement du produit et des ressources/outils, le déchargement et le nettoyage de la station [Burns and Daganzo, 1987]. Le processus de changement de séries des stations est coûteux vu la consommation d'énergie additionnelle et la présence d'opérateur, ce qui augmente le coût de fonctionnement d'une ligne flexible ou reconfigurable. En outre, les temps de changement de séries influencent la productivité de lignes en augmentant le temps de production. Ces aspects doivent être pris en compte lors de conception d'une telle ligne.

La Figure 1.2 montre la comparaison de trois types de lignes de fabrication selon trois critères : le coût d'installation et de fonctionnement, le taux de production et le niveau de variabilité. Le constructeur d'une ligne de fabrication doit décider quel type convient le mieux. En premier lieu, il doit tenir compte de facteurs liés à la demande produits, à savoir la variabilité de produits à fabriquer, le volume de production nécessaire et la durée supposée de demande. En outre, il doit prendre en considération le facteur de coût d'installation et de fonctionnement de ligne qui inclut le coût des ressources, les coûts de changements de séries et d'exploitation etc.

Avant de classer des lignes de fabrication en utilisant le critère de flux de produits il faut donner quelques définitions importantes.

La **charge d'une station** est l'ensemble des opérations affectées à une station. Ainsi, le

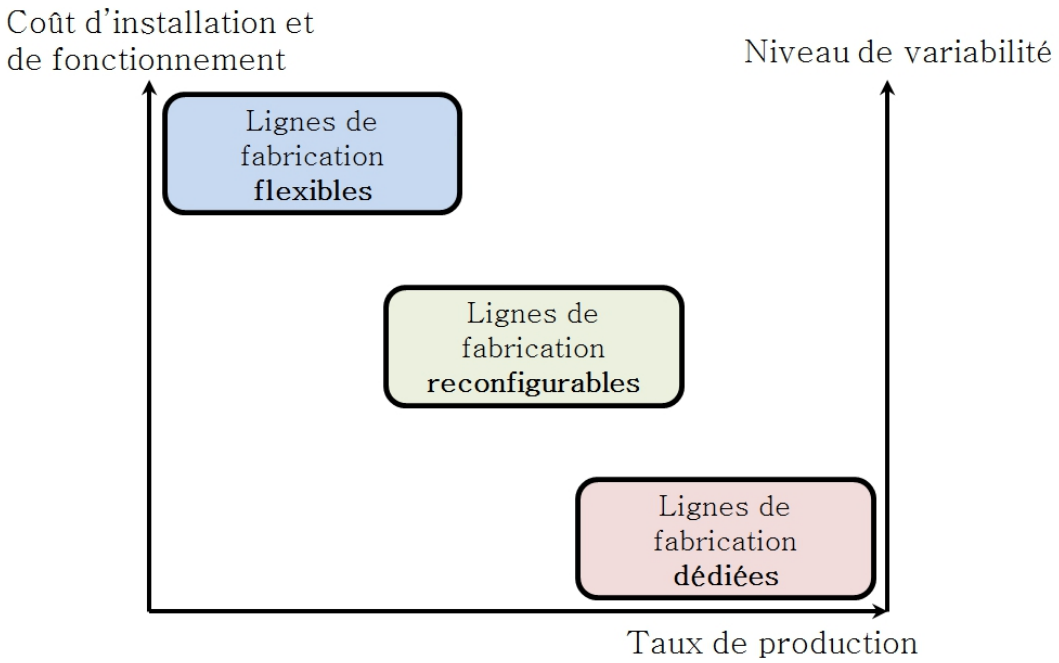


FIGURE 1.2 – Comparaison de trois types de lignes de fabrication.

montant de travail nécessaire pour transformer des produits bruts en produits finis se divise en charges de stations qui, à leur tour, se divisent en opérations indivisibles.

Le **temps opératoire** est le temps d'exécution d'une opération. Il est indispensable de prendre en compte ce paramètre de temps dans le processus de conception de lignes car il influence leur productivité.

Le **temps de transport** de produits. C'est le temps dépensé pour le transport des unités de produits entre les stations par un dispositif de transfert qui peut être un tapis ou un pont roulant par exemple. Ce temps est normalement stable et ne dépend pas des stations. Pourtant, il est possible que certaines stations puissent causer des temps additionnels de transport et cela doit être considéré lors de la conception de lignes de fabrication [Bard, 1989].

Le **temps de station** est le temps d'exécution de la charge de station. Il est égal à la somme de temps opératoires en cas d'exécution séquentielle des opérations sur les stations et au temps opératoire le plus élevé en cas d'exécution parallèle.

Le **temps d'inactivité** (ou temps mort) est le temps pendant lequel une station est inactive, c'est-à-dire, les opérations affectées à cette station ne sont pas effectuées.

Le **temps de cycle**. Il y a deux définitions de temps de cycle.

La *première définition* décrit le temps de cycle comme le temps de fabrication moyen d'une unité de produit d'un certain type. Soit c le temps de cycle, t_a le temps d'achèvement de fabrication de la dernière unité de produit, u - le nombre de produits à fabriquer. Le temps de cycle est calculé de la façon suivante :

$$c = \frac{t_a}{u} \tag{1.1}$$

Cette définition est très proche du terme « takt time » qui signifie un rapport entre le temps accessible de la période de production et la demande dans cette période. Dans ce cas, c'est une condition qui doit être vérifiée lors de la configuration d'une ligne de fabrication.

Si le *débit de ligne* (*throughput* en anglais) qui signifie le nombre de produits à fabriquer par unité de temps est donné, on peut calculer le temps de cycle comme suit : $c = \frac{1}{throughput}$. Le débit est donc égal à l'inverse du temps de cycle : moins le temps de cycle est élevé plus le débit est important. Évidemment, le débit de ligne est identique à la notion de productivité.

La *deuxième définition* détermine le temps de cycle comme le temps qui sépare les sorties de deux produits finis consécutifs. Cette définition s'applique uniquement aux environnements de production où les produits du même type sont fabriqués l'un après l'autre sans autres types de produits entre eux. Dans le cas des lignes dédiées cette définition est valide parce que tous les produits sont de même type.

Afin d'illustrer la notion de temps de cycle, nous présentons la Figure 1.3 avec un exemple d'une ligne de fabrication simplifiée. Cette ligne se compose de 3 stations. Le cheminement de production est unique. La ligne est dédiée. Les temps de station en minutes sont indiqués au-dessus des stations. Le temps de transport est négligeable.

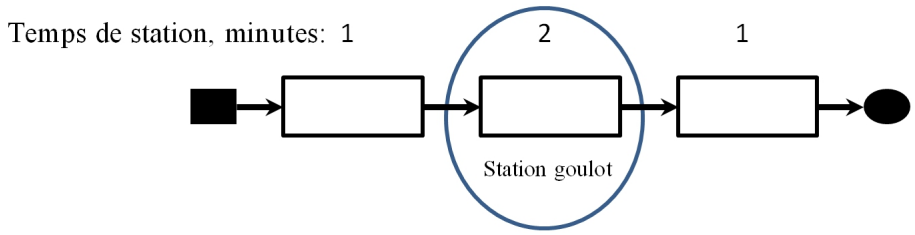


FIGURE 1.3 – Temps de cycle.

Considérons la première définition du temps de cycle. Soit le nombre de produits à fabriquer $u = 3$. Le processus de production commence au temps zéro. La fabrication du premier produit

se finira évidemment dans 4 minutes. Le deuxième produit doit d'abord attendre 1 minute le traitement du premier produit à la première station. Ensuite il est traité à la première station, tandis que le premier produit est traité à la deuxième station. Vu que le temps de la deuxième station est égal à 2 minutes, le deuxième produit doit attendre encore 1 minute à la première station pour que le traitement du premier produit soit fini. Ainsi, le temps d'achèvement du deuxième produit sera égal à 6 minutes. L'écart de 2 minutes entre deux produits est déterminé par le temps d'attente du traitement. La même logique s'applique au troisième produit dont le temps d'achèvement sera égal à 8 minutes (2 minutes d'attente de plus). Donc, le temps de cycle peut être calculé selon la formule de la première définition de temps de cycle :

$$c = \frac{t_a}{u} = \frac{8}{3} = 2,66$$

Selon la deuxième définition, le temps de cycle est égal au temps entre les sorties de deux produits consécutifs. Dans ce cas, il est égal à 2 ($6 - 4$ pour le deuxième et le premier produits et $8 - 6$ pour le troisième et le deuxième produits).

Dans le chapitre 3 nous considérons une ligne de fabrication dédiée qui fabrique un seul type de produits. Dans le chapitre 4 il y a une borne commune pour les temps de cycles de tous les types de produits, et nous supposons que cette borne n'est dépassée par aucun temps de cycle. Dans le chapitre 5 nous examinons le cas d'une ligne multi-produit pour laquelle les produits sont groupés en lots de produits identiques. Autrement dit, la séquence de produits est telle que dans un lot de produits d'un certain type il n'y a pas de produits d'autres types. Ces conditions permettent d'appliquer la deuxième définition de temps de cycle. En effet, le temps de cycle (qui représente donc le temps entre les sorties de deux produits consécutifs) est égal au temps de station la plus chargée. Afin de le prouver prenons notre exemple. Si, le nombre de produits était égal à 100, le temps d'achèvement du dernier produit serait égal à 202 minutes. Ainsi, on peut calculer le temps de cycle selon la formule 1.1 de la première définition de temps de cycle :

$$c = \frac{t_a}{u} = \frac{202}{100} = 2,02$$

Cette valeur est très proche de la valeur de temps de la deuxième station. Effectivement, cette valeur s'approche asymptotiquement de 2 :

$$\lim_{u \to \infty} \frac{4+2(u-1)}{u} = 2$$

Le débit de ligne est respectivement égal à 0,5 produits par minute :

$$Throughput = \frac{1}{c} = \frac{1}{2}$$

La station dont le temps est maximal est appelée une *station goulot* (*bottleneck station* en anglais). Dans cet exemple, le temps de cycle est égal à 2 minutes car c'est le temps de la station goulot qui est la deuxième station.

En ayant clarifié une notion de temps de cycle, on peut maintenant classifier les lignes de fabrication suivant le critère de **flux de produits** :

1. Les *lignes de fabrication cadencées* correspondent aux lignes dont les temps de stations sont bornés par le temps de cycle. Des produits bruts sont transportés le long d'une ligne de fabrication avec une cadence constante déterminée par le temps de cycle. Tous les produits qui se trouvent à un moment donné sur toutes les stations sont simultanément transportés vers les stations suivantes. Vu que les opérations sont indivisibles, le temps de cycle ne peut pas être plus petit que le plus grand temps d'opération. Le temps d'inactivité apparaît si le temps de certaines stations est moins grand que le temps de cycle. Évidemment, la station dont le temps est égal au temps de cycle n'a pas de temps d'inactivité. Autrement dit, pour chaque station le temps de station plus le temps d'inactivité doivent être égal au temps de cycle.

2. Les *lignes de fabrication non-cadencées* se distinguent des lignes cadencées par la présence de *stocks tampons* dans lesquels une quantité limitée de produits semi-finis est gardée lorsque la station suivante traite encore les produits bruts précédents. L'objectif des stocks tampons consiste à pallier aux malfonctionnements de la ligne comme, par exemple, les défaillances de machines. Ici, le **temps de cycle** est défini comme le temps moyen de la station la plus chargée parce que la cadence de la ligne n'est pas constante. Quant au temps d'inactivité, en dehors de la différence entre le temps de cycle et le temps de station, il peut être causé par deux facteurs liés aux stocks tampon : soit le stock tampon suivant est plein soit le stock tampon précédent est vide parce que la station précédente n'a pas encore fini l'exécution de sa charge [Scholl, 1999, Dolgui et al., 2002].

1.3 La conception préliminaire des lignes de fabrication

Il y a des entreprises dont la spécialité est la conception et la production de lignes de fabrication. Normalement, un fabricant de lignes (équipementier) reçoit de son client l'information à propos des produits que ce dernier souhaite fabriquer : les formes géométriques, les matériaux constitutifs, les types de produits avec leurs spécifications et les volumes annuels de production souhaités. Ensuite l'équipementier doit fournir une ou plusieurs solutions préliminaires concernant l'architecture de la ligne, le nombre de stations, les ressources utilisées et naturellement une estimation du coût de la ligne. Vu la grande concurrence dans ce secteur et les exigences accrues de la part des clients, la solution doit d'un côté être fournie au plus vite et d'un autre côté être de la meilleure qualité possible. Par une bonne qualité on sous-entend la satisfaction de critères imposés par le client, notamment le coût, la productivité de la ligne et sa fiabilité. L'optimisation à cette étape est très importante car la signature ou non d'un contrat de fabrication d'une telle ligne en dépend et les valeurs de contrat se chiffrent parfois en millions d'euros [Delorme et al., 2009].

Donc avant qu'une ligne de fabrication soit produite et assemblée, une procedure scrupuleuse de conception préliminaire s'impose. La conception préliminaire est un processus qui a pour but de définir une configuration de la ligne de fabrication et de conduire à sa mise en œuvre [Dolgui and Proth, 2006]. Le problème de conception préliminaire se divise en plusieurs étapes qui doivent être effectuées dans l'ordre chronologique suivant :

1. **Analyse de produit**. Cette étape consiste en l'examen méticuleux des produits qui seront fabriqués afin de déterminer l'ensemble des opérations que la ligne doit effectuer. Chaque opération doit être bien précisée et décrite. Les outils de la *Conception Assisté par Ordinateur* (CAO), plus connue sous le terme anglais Computer Aided Design ou CAD, représente une technologie efficace qui aide à développer un prototype de produit avec ses propriétés : les composants, les dimensions géométriques et la typologie.
2. **Planification de processus** ou choix de gammes. À ce stade les gammes possibles de fabrication sont choisies définissant les contraintes technologiques. Cela inclut, par exemple, l'ordre partiel entre les opérations, les contraintes d'inclusion et d'exclusion

entre les opérations, etc. En plus, une sélection d'architecture de ligne doit être effectuée. Le résultat de cette sélection définit les cheminements de production.

3. **Configuration**. Cette phase consiste au choix des ressources et à leur affectation à des stations afin de remplir les objectifs donnés par le client, par exemple, assurer la productivité de ligne souhaitée. Les ressources sont affectées aux stations tout en respectant les contraintes technologiques et en tenant compte de l'architecture de ligne qui sont établies à l'étape précédente.
4. **Analyse de flux**. Cette étape s'intéresse à la simulation utilisée pour l'examen de flux de produits le long de la ligne en tenant compte de la variabilité de produits et des événements aléatoires. L'objectif est de dimensionner un dispositif de transfert et de trouver le placement optimal des ressources qui ont été affectées aux stations lors des étapes précédentes.
5. **Ordonnancement**. À ce stade l'ordre d'entrée des produits sur la ligne de fabrication est défini. Le but de l'ordonnancement est d'utiliser au mieux les ressources affectées.

Si nécessaire, le concepteur revient aux étapes précédentes afin de corriger ou réviser les decisions déjà prises. Dans le cas de la conception d'une ligne reconfigurable, le processus de conception peut recommencer dès le début après une décision de fabriquer un nouveau produit. Cette méthodologie a déjà été mise en œuvre dans un logiciel d'aide à la décision développé pour la conception préliminaire de lignes dédiées [Battaïa et al., 2012].

Cette thèse est consacrée à une étape importante de la conception préliminaire : la configuration de lignes de fabrication. Les décisions prises lors de cette étape influencent fortement les objectifs définis par le client, c'est-à-dire, le coût d'installation et de fonctionnement, le taux de production et la fiabilité de la ligne.

1.4 Conclusion

Dans ce chapitre nous avons donné une définition des lignes de fabrication et mis en évidence leurs caractéristiques. Une ligne de fabrication est un système de production à flux continu, destiné à la fabrication de produits finis à partir de produits bruts. Elle se compose de stations

alignées de manière séquentielle. Les lignes de fabrication correspondent à deux types de systèmes de production : le système d'ateliers à cheminement unique et le système de production hybride. La différence entre ces derniers est dans le nombre de cheminements possibles de production. Pourtant, quelque soit ce nombre, la direction de flux est unique, ce qui les distingue de systèmes d'ateliers à cheminement multiple caractérisés par de multiples directions de flux de produits. Ensuite, nous avons proposé une classification des lignes de fabrication suivant les critères de variabilité de production et de flux de produits. En utilisant critère de variabilité nous avons distingué les lignes dédiées, flexibles et reconfigurables. La classification suivant le critère de flux de produits fait apparaître les lignes cadencées et non-cadencées. Quelques définitions de base ont également été données afin de préparer le lecteur à une meilleure compréhension des chapitres suivants. Parmi d'autres, les notions de changement de séries et de temps de cycle ont été introduites. Elles sont très importantes pour la description des problèmes étudiés. Un changement de séries est l'ensemble de démarches nécessaires pour préparer une station au traitement d'un produit brut d'un certain type dans le cas de lignes flexibles ou reconfigurables. Le changement de séries suscite un accroissement du coût de la ligne et du temps de production. C'est pourquoi il est important de considérer cet aspect. Le temps de cycle, un autre élément sur lequel nous voulons concentrer l'attention de lecteur, signifie le temps de fabrication moyen d'une unité de produit d'un certain type. Dans le cas de fabrication successive de produits de même type sur la ligne cadencée, il est égal au temps entre les sorties de deux produits consécutifs. Ce dernier temps, à son tour, est égal au temps de la station la plus chargée. Cette station est appelée station goulot. Le temps de cycle influence directement la productivité de la ligne. Nous avons finalement indiqué le positionnement du problème de configuration de lignes de fabrication parmi les autres problèmes complexes de la conception préliminaire de lignes de fabrication. Dans le chapitre 2 nous allons nous concentrer sur les problèmes de configuration de lignes de fabrication et nous allons présenter un état de l'art des principaux travaux développés pour cette problématique.

Chapitre 2

État de l'art sur les problèmes de configuration de lignes de fabrication

2.1 Choix de ressources et équilibrage de lignes de fabrication

Les problèmes que nous étudions dans cette thèse correspondent à la troisième étape de la conception préliminaire d'une ligne de fabrication, à savoir, le choix de sa configuration. À ce stade toutes les nuances technologiques sont déjà prises en compte. Les produits à fabriquer et leurs caractéristiques détaillées sont connus. Les opérations nécessaires pour la transformation des produits bruts en produits finis ont été définies. L'ordre partiel entre les opérations est donné et peut être répresenté sous forme d'un graphe. Les contraintes technologiques réduisant l'espace des solutions possibles à ce stade sont établies.

Dans le chapitre précedent nous avons mentionné que la configuration consiste en deux tâches : le choix des ressources qui exécutent les opérations et l'affectation des ressources à des stations. Ces deux tâches sont effectuées selon les objectifs donnés. Les contraintes technologiques doivent forcément être prises en compte sinon la solution fournie sera inadmissible. Dans la plupart des publications, le problème de **choix des ressources** n'est pas pris en compte pour la configuration de lignes de fabrication. Les auteurs supposent que les ressources sont connues. Il y a deux situations dans lesquelles cette hypothèse est justifiée : soit la sélection est triviale, soit les ressources sont créées au stade de l'analyse de produits (à partir du moment où on connaît les opérations, on connaît aussi les ressources). Pourtant, le choix des ressources n'est pas toujours évident. Si les opérations peuvent être exécutées

par plusieurs ressources, par exemple, par plusieurs équipements disponibles sur le marché, alors la sélection des ressources devient une tâche difficile qui nécessite des méthodes de résolution combinatoires [Pinto et al., 1983, Bukchin and Tzur, 2000, Bukchin and Rubinovitz, 2003, Nicosia et al., 2002, Dolgui and Ihnatsenka, 2009a, Li et al., 2011]. Le cas où le choix de ressources et leur affectation sont indissociables reste cependant relativement peu traité dans la littérature. Dans le chapitre 3 nous présentons un problème de ce type pour une ligne dédiée.

Quant aux problèmes d'affectation des ressources, ils ont bénéficié de beaucoup plus d'attention de la part des chercheurs. Le processus d'affectation des opérations et des ressources à des stations est appelé l'équilibrage d'une ligne de fabrication. Nous proposons la définition suivante. L'**équilibrage d'une ligne de fabrication** (connu par un terme anglais *assembly line balancing* ou simplement ALB) est une affectation admissible des ressources qui exécutent les opérations à des stations de sorte que chaque opération appartient à exactement une station, les contraintes technologiques sont respectées et aucun temps de station, ou un temps moyen de station, n'excède le temps de cycle. Globalement, les problèmes d'équilibrage de lignes de fabrication ont généré un grand nombre de problèmes spécifiques d'optimisation combinatoire.

2.2 Les problèmes d'équilibrage de lignes de fabrication

Les industries utilisant des lignes de fabrication peuvent profiter de leur équilibrage optimal. Cela se manifeste par la réduction des coûts, l'accroissement de la productivité et l'augmentation de leur rentabilité. Par conséquent, la compétitivité des industries qui recourent aux méthodes d'optimisation s'accroît. L'équilibrage optimal favorise non seulement les producteurs en augmentant leur compétitivité sur le marché, mais influence aussi positivement la satisfaction de la demande. C'est pourquoi les problèmes d'équilibrage de lignes de fabrication attirent l'attention de nombreux chercheurs qui, dans leurs travaux, utilisent différentes méthodes de recherche opérationnelle afin de trouver une solution optimale ou quasi optimale dans les cas où la taille du problème ou le temps de résolution imparti ne permettraient pas une solution optimale.

Plusieurs classifications des problèmes ALB ont été proposées dans la littérature [Baybars, 1986, Rekiek et al., 2002, Boysen et al., 2007, Guschinskaya and Dolgui, 2010]. Nous proposons ici de les classer suivant les six éléments qui permettent de bien définir les problèmes

abordés dans cette thèse :

1. La **variabilité de ligne**. Ici, nous utilisons la classification de lignes de fabrication du chapitre 1, à savoir, nous distinguons des lignes dédiées, flexibles et reconfigurables.
2. Les **caractéristiques de ligne**. Dans cette catégorie nous incluons l'architecture de ligne, c'est-à-dire, le nombre de cheminements de production et le schéma logique de ligne. Par ailleurs, le cadencement de flux de produits est défini. Cela nous renvoie aux notions de lignes de fabrication cadencées et non-cadencées.
3. Les **caractéristiques de temps**. Nous décrivons les élements de la ligne de fabrication liés à la dimension du temps : les temps opératoires, les temps de stations, les temps de changements de séries, le temps de cycle, etc.
4. Les **caractéristiques de coût**. Les élements de la ligne de fabrication sont considérés du point de vue de la dimension de coût. Par exemple, nous distinguons les coûts de ressources, les coûts de changements de séries, les coûts d'installation de stations, etc.
5. Les **contraintes technologiques**. Les restrictions imposées par des facteurs technologiques sont mathématiquement définies à ce stade.
6. Les **critères d'optimisation**. Chaque problème d'équilibrage est caractérisé par une ou plusieurs fonctions objectif devant être minimisées ou maximisées.

Ainsi, cette classification nous permet de distinguer des lignes selon leurs fréquences de changements de types de produits fabriqués (notion de variabilité) et selon les temps et les coûts à prendre en compte ainsi que les contraintes technologiques à considérer et les fonctions objectif à optimiser.

L'histoire des publications sur l'équilibrage de lignes de fabrication tient son origine dans le travail de Salveson publié en 1955 [Salveson, 1955]. Cette étude est concentré sur l'aspect principal de la configuration de lignes, à savoir, sur l'affectation de ressources et, effectivement, d'opérations à des stations. Plus tard, en 1986, le problème traité par Salveson a été baptisé *Simple Assembly Line Balancing Problem* (SALBP) par Baybars [Baybars, 1986].

C'est le problème de base qui est devenu le point de départ pour tout un ensemble des problèmes de conception de lignes. En utilisant notre classification, on peut décrire le SALBP de la manière suivante :

1. La *variabilité de ligne*.
   - Un seul type de produit est fabriqué et la ligne est dédiée.
2. Les *caractéristiques de ligne*.
   - Il y a un seul cheminement de production et il n'y a pas de stations parallèles. Les stations $k = 1, \ldots, m$ sont alignées de manière sérielle l'une après l'autre.
   - La ligne est cadencée. Les produits bruts restent pendant un temps constant sur chaque station qui est égal au temps de cycle c. Ainsi, la ligne fonctionne avec la productivité $1/c$ fixe.
3. Les *caractéristiques de temps*.
   - Les temps opératoires $p_i, i = 1, \ldots, n$ sont déterministes.
   - Le temps de cycle fixe est donné.
4. Les *caractéristiques de coût*.
   - La dimension de coût n'est pas considérée. On suppose qu'en minimisant le temps d'inactivité on minimise le coût.
5. Les *contraintes technologiques*.
   - L'exécution de chaque opération doit se passer sur une seule station.
   - Chaque station peut être également équipée de sorte qu'elle puisse effectuer n'importe quelle opération et que n'importe quelle opération $i = 1, \ldots, n$ puisse être exécutée sur n'importe quelle station.
   - Les temps de stations ne doivent pas dépasser le temps de cycle.
   - Les contraintes de précédence sont imposées à certaines opérations selon le processus technologique choisi.
   - Le mode d'exécution des opérations est séquentiel. Le temps de station est égal à la somme des temps opératoires des opérations affectées à cette station.
6. Les *critères d'optimisation*.
   - L'objectif est de minimiser le nombre de stations.

Plus précisément, le problème initial de Salveson a été désigné SALBP-1. Trois autres formulations basées sur les mêmes hypothèses, mais avec un critère d'optimisation différent ont aussi été définies : SALBP-2, SALBP-E et SALBP-F.

Tableau 2.1 – Les formulations de SALBP

Nombre de stations, m	Temps de cycle, c	
	Donné	À minimiser
Donné	SALBP-F	SALBP-2
À minimiser	SALBP-1	SALBP-E

Le Tableau 2.1 montre que les distinctions entre différentes versions du SALBP se trouvent dans le rapport « temps de cycle / nombre de stations ». Les trois types de contraintes : l'indivisibilité des opérations, la limitation des temps de station par le temps de cycle et les contraintes de précédence, sont inhérentes à toute version du SALBP.

- Le **SALBP-F** est le problème de faisabilité. Ici, le temps de cycle ainsi que le nombre de stations servent comme des paramètres donnés. La solution de ce problème consiste à répondre à la question : est-il possible de répartir l'ensemble d'opérations N sur m stations en respectant toutes les contraintes du SALBP. Autrement dit, il faut décider si une ligne de fabrication à m stations peut fonctionner avec le temps de cycle c défini.
- Pour le **SALBP-1** le temps de cycle est donné comme un paramètre alors que le nombre de stations est variable et doit être minimisé.
- Le **SALBP-2** est un problème inverse au SALBP-1. Il consiste à la minimisation du temps de cycle ayant le nombre de stations fixe.
- Le **SALBP-E** a la fonction objectif non-linéaire qui répresente l'efficacité de la ligne - le produit du nombre de stations et du temps de cycle $m \cdot c$. En minimisant ce produit, on maximise l'efficacité.

Au regard de la théorie de la complexité, le SALBP-1 est un problème NP-difficile. Wee et Magazine [Wee and Magazine, 1982] ont démontré ceci en se basant sur le problème bin packing qui a déjà été prouvé d'être NP-difficile [Garey and Johnson, 1979]. Il s'agit de trouver le rangement le plus efficace pour un ensemble d'objets unidimensionnels de valeurs différentes dans les boîtes de capacités données. Dans ce cas, les objets correspondent à des opérations ayant une seule dimension - le temps opératoire, tandis que les boîtes sont des stations. Si le temps de cycle est la capacité maximale de boîtes, alors le problème consiste à distribuer les

opérations entre les stations de sorte qu'aucun temps de station ne dépasse le temps de cycle et que le nombre de stations soit minimisé. En ayant des séquences d'opérations à respecter, c'est-à-dire des contraintes de précédence, le problème bin packing devient SALBP-1. Quant aux autres versions du SALBP, elles ont été également démontré NP-difficiles en utilisant le problème NP-difficile d'ordonnancement sur m machines identiques parallèles pour SALBP-2 ou le problème de partition pour SALBP-F. Le SALBP-E est évidemment NP-difficile lui aussi puisqu'il représente une généralisation des autres problèmes du type SALBP.

De nombreuses études bibliographiques ont été faites dont les plus récentes sont [Erel and Sarin, 1998, Rekiek et al., 2002, Becker and Scholl, 2006, Guschinskaya and Dolgui, 2010]. Une description détaillée de règles de dominance permettant de réduire l'effort calculatoire ainsi que de calculs de bornes inférieures et supérieures visant à la réduction de l'espace de recherche et à l'estimation de la qualité des solutions approchées pour les SALBP est donnée dans [Scholl, 1999].

Même si la majorité écrasante des travaux de recherche dans le domaine de l'équilibrage de lignes est consacrée aux SALBP, les problèmes traités ont intégré d'autres aspects plus complexes nous amenant à différentes généralisations. Cela a rapproché des travaux académiques avec les situations pratiques que l'on trouve dans l'industrie. Au fil du temps, en essayant d'approcher les problèmes initiaux à des conditions réelles, les chercheurs ont enlevé certaines hypothèses simplificatrices du SALBP en introduisant de nouvelles hypothèses plus réalistes. Le terme **GALBP** (pour Generalized Assembly Line Balancing Problem) a été proposé par Baybars [Baybars, 1986] pour distinguer les problèmes ayant au moins une hypothèse plus générale par rapport au SALBP. Des études montrant la diversité d'hypothèses et de méthodes de résolution pour les GALBP ont été présentées par [Ghosh and Gagnon, 1989, Rekiek et al., 2002, Becker and Scholl, 2006].

2.3 Principales caractéristiques des problèmes d'équilibrage de lignes de fabrication

Afin de présenter les études existantes consacrées aux problèmes d'équilibrage de lignes de fabrication, nous proposons de suivre notre classification des lignes de fabrication. Nous allons

parcourir les élements de cette classification et analyser les hypothèses qui sont utilisées dans la littérature.

2.3.1 Variabilité de lignes

Le problème d'équilibrage est inhérent à tous les types de lignes quelque soit le critère de variabilité.

La plupart des publications sont naturellement concentrées sur les *lignes dédiées*. Rappelons que les problèmes de type SALBP viennent des lignes dédiées. Parmi les nombreuses études consacrées au SALBP on peut citer les travaux les plus récents : [Blum, 2008, Pastor and Ferrer, 2009, Wei and Chao, 2011, Kilincci, 2011]. L'hypothèse d'un seul type de produits est très répandue. Elle est également utilisée dans de nombreux modèles plus généraux, voir par exemple [Sarin et al., 1999, Gadidov and Wilhelm, 2000, Sabuncuoglu et al., 2009, Dolgui and Ihnatsenka, 2009b, Hamta et al., 2011].

L'équilibrage de *lignes multi-produits, flexibles ou reconfigurables*, destinées à la fabrication d'une famille de produits, attire également l'attention de beaucoup de chercheurs. Dans le chapitre 1, nous avons mentionné que la séquence de stations est la même pour tous les types de produits. Cependant, la séquence d'opérations diffère selon le type de produits. Dans ce cas, certaines stations peuvent être inactives. Les graphes de précédence de chaque produit doivent être simultanément pris en compte lors de l'équilibrage. Le lancement d'un produit d'un certain type sur la ligne suscite la préparation des stations qui peut nécessiter un certain temps et/ou coût. [Boysen et al., 2007] distinguent les lignes modèles-mixtes (mixed-model lines) et les lignes multi-modèles. La différence entre ces deux cas réside dans les changements de séries. Pour les lignes modèles-mixtes, les processus de fabrication de différents produits sont tellement similaires que les temps de changements de séries peuvent être négligés. Ainsi, les différents produits peuvent être fabriqués dans un ordre arbitraire. Nous pouvons citer quelques travaux recents liés à l'équilibrage de lignes modèles-mixtes : [Yagmahan, 2011, Xu and Xiao, 2011, Rabbani et al., 2012]. Quant aux lignes multi-modèles, le niveau de variabilité de produits est plus élevé et la prise en compte de changements de série devient nécessaire. Afin de réduire les temps et les coûts de changements de série, les produits sont regroupés en lots. La taille et la

séquence de lots font ainsi partie de la décision [Chakravarty and Shtub, 1985, Kimms, 2000, Pastor et al., 2002].

Depuis l'introduction de la notion de *systèmes de production reconfigurables* par Koren et al. [Koren et al., 1999] très peu d'attention a été accordée à l'équilibrage de ces systèmes. Nous pouvons citer [Son et al., 2001, Essafi et al., 2012, Musharavati and Hamouda, 2012, Wang and Koren, 2012]. Le manque de travaux de recherches dans ce domaine est lié à la nouveauté des systèmes reconfigurables et à leur rareté dans l'industrie par rapport aux autres systèmes de production. Notons pourtant que leur fonctionnement peut être assimilé à celui des lignes flexibles multi-modèle (par lot) à la seule différence que le changement de série devient encore plus coûteux car il demande une reconfiguration physique de la ligne.

2.3.2 Caractéristiques de lignes

Nous considérons les deux caractéristiques suivantes : l'architecture de la ligne et le type de flux de produits bruts.

Le choix d'*architecture d'une ligne* implique un nombre donné de cheminements de production et un schéma logique de ligne.

Quand on ajoute au moins une station parallèle, cela donne la possibilité d'avoir un autre cheminement de production. Les stations parallèles ont les mêmes types de ressources affectés. Elles reçoivent alternativement des produits pour leur traitement. L'installation de stations parallèles a pour but de diminuer le risque de pannes ou d'augmenter le temps de cycle local si certaines opérations dépassent le temps de cycle souhaité [Bard, 1989, Daganzo and Blumenfeld, 1994, Ege et al., 2009]. La parallèlisation (duplication) peut également s'appliquer au niveau des lignes [Süer, 1998, Gökçen et al., 2006, Scholl and Boysen, 2009]. La mise en place de lignes parallèles augmente la productivité et la fiabilité du système de production. Cependant, leur coût d'installation est plus élevé par rapport à des lignes avec un seul cheminement de production.

Le schéma logique de ligne avec des stations installées les unes après les autres n'est pas le seul qui existe. Les principes de « *just-in-time* » poussent les industriels à recourir à des lignes en U. Une ligne en U implique que certaines stations traitent le même produit lors de

stades différents du processus de fabrication. De nombreux chercheurs s'intéressent à l'équilibrage de lignes en U, voir par exemple [Miltenburg and Wijngaard, 1994, Hwang et al., 2008, Sabuncuoglu et al., 2009, Rabbani et al., 2012].

En tenant compte du type de *flux de produits* utilisé, on distingue les lignes cadencées et les lignes non-cadencées (voir le chapitre 1). Les approches d'équilibrage de ces deux types de lignes sont différentes, étant donné la présence de stocks tampons dans les lignes non-cadencées. La plupart des travaux sont consacrés aux lignes cadencées, y compris, bien évidemment, les articles sur les SALBP. Une étude bibliographique exhaustive sur l'équilibrage de lignes cadencées a été proposé par [Guschinskaya and Dolgui, 2010]. La configuration de lignes non-cadencées est plus complexe vu la nécessité de résoudre deux problèmes supplémentaires en dehors de l'équilibrage, à savoir : l'affectation optimale de stocks tampons et l'estimation de la productivité de la ligne. Optimiser à la fois l'affectation des opérations et les stocks tampon est une tâche très difficile. Dans la majorité des publications l'équilibrage est donné, tandis que le choix de la disposition et de la capacité des stocks tampons optimisant un critère donné est recherché, voir par exemple [Lutz et al., 1998, Dolgui et al., 2002, Dolgui et al., 2007, Shi and Gershwin, 2009, Massim et al., 2010].

2.3.3 Caractéristiques de temps

Comme nous l'avons constaté dans le chapitre 1, les temps opératoires sont très importants. En fonction des temps opératoires on calcule les temps de stations et vérifie la contrainte sur le temps de cycle. Les temps d'inactivité dépendent donc directement des valeurs des temps opératoires et du temps de cycle.

Dans les publications existantes, les temps opératoires sont représentés de trois manières différentes.

Temps opératoire déterministe. Cette hypothèse est utilisée lorsque la variabilité de temps d'exécution d'opération est suffisamment petite. L'emploi de ce type de temps opératoires convient essentiellement aux lignes de fabrication automatisées. Si le processus de fabrication est manuel, les temps opératoires peuvent rester déterministes seulement dans le cas du travail d'ouvriers hautement qualifiés et motivés [Rekiek et al., 2002]. Pourtant l'hypothèse de temps

déterministe est très souvent utilisée dans la littérature, voir l'état de l'art de Boysen et al. [Boysen et al., 2007].

Temps opératoire stochastique. Les temps opératoires suivent une distribution de probabilités connue ou même inconnue/partiellement connue [Boysen et al., 2007]. L'incertitude sur les temps opératoires vient du fait que les niveaux de compétence et de motivations des opérateurs ne sont pas homogènes. Concernant, les machines automatiques, la probabilité de panne et de malfonctionnement apporte le facteur d'incertitude dans la définition des temps opératoires. De nombreuses études dans des contextes différents ont été consacrées aux problèmes liés à l'équilibrage de lignes avec des temps opératoires stochastiques, voir par exemple [Moodie and Young, 1965, Kottas and Lau, 1976, Buzacott and Shanthikumar, 1993, Wild, 1998, Sarin et al., 1999, Khan and Day, 2002, Tiacci, 2012].

Temps opératoire dynamique. Ce type de temps apparaît seulement dans le cas de fabrication manuelle. Le changement de temps opératoire dépend des effets d'apprentissage et d'oubli des ouvriers. Les temps opératoires sont normalement plus grands en début de fonctionnement d'une ligne de fabrication parce que les ouvriers ne sont pas encore habitués aux nouveaux postes de travail. Ils se réduisent au fur et à mesure grâce à l'apprentissage et l'habitude. Pourtant, l'effet d'oubli qui surgit en raison des pauses de travail, peut les augmenter de nouveau. Le taux de réduction/augmentation des temps opératoires dépend du type d'opérations effectuées, du degré de variabilité de la fabrication et naturellement du niveau de qualification des opérateurs. Les formulations avec des temps dynamiques et des effets d'apprentissage et d'oubli ont été étudiées, par exemple, dans [Thomopoulos and Lehman, 1969, Chakravarty, 1988, Shafer et al., 2001]. En outre, un effet de déterioration peut aussi apparaître. La répétition continue de tâches similaires entraîne la fatigue des ouvriers, et, comme conséquence, l'augmentation des temps opératoires et une détérioration possible de la qualité de produits, voir [Duran Toksari et al., 2010, Karimi-Nasab et al., 2012].

Outre les temps opératoires, les *temps de changement de série* influencent fortement la productivité d'une ligne de fabrication. Chaque changement de séries exige un certain temps de préparation des stations pour le traitement d'un nouveau produit [Yokoyama, 2008]. Cela augmente le temps de séjour d'un produit sur la ligne. De plus, le temps de changement de séries

peut varier en fonction de la séquence d'opérations [Andrès et al., 2008, Scholl et al., 2008]. Ainsi, le temps de changement de séries doit être pris en compte lors du calcul du temps de cycle.

2.3.4 Caractéristiques de coût

Les coûts jouent un rôle important. Vu que nous considérons la conception préliminaire de lignes de fabrication, nous nous intéressons en premier lieu aux *coûts d'installation* qui définissent le montant d'investissement nécessaire pour mettre en place une ligne. Les coûts d'installation incluent :

- Le *coût d'équipement*. Il comprend soit le prix d'achat d'équipement, soit une estimation de son coût de fabrication.
- Le *coût du capital*. Ce coût est lié à chaque station. Il a été introduit par Amen [Amen, 2000] qui a supposé que ce coût dépend de la longueur de la ligne en termes de nombre de stations. Donc, c'est une sorte de coût d'ouverture de station. Ce coût est souvent considéré étant fixe.

Hormis les coûts d'installation, il y a de nombreux *coûts variables* qui apparaissent après l'installation d'une ligne. Nous allons mentionner les plus importants :

- Le *coût de fonctionnement d'équipement*. Dans cette catégorie nous rangeons les amortissements, le coût de maintenance et de réparation et les taux d'intérêt. Ce coût dépend du nombre de stations et du type d'équipement utilisé [Scholl, 1999].
- Le *coût salarial*. Le montant de salaires dépendent du nombre d'opérateurs, de leur qualification et du type de travail à réaliser. Des formulations de problèmes d'équilibrage de lignes de fabrication qui tiennent compte des salaires ont été proposées par [Rosenberg and Ziegler, 1992, McMullen and Frazier, 1998, Amen, 2001].
- Le *coût de stockage*. Dans le cas de lignes non-cadencées les produits sortent des stations dès que le traitement est fini. Si la station suivante est occupée, alors le produit reste dans un stock tampon. Lorsqu'il reste en stock, le bénéfice se réduit car il y a moins de produits fabriqués et il y a une immobilisation de ressources financières liées à la valeur du produit. Cette perte de bénéfice peut être répresentée en tant que

coût de stockage. Les coûts de stockage sont pris en compte dans certains travaux, voir [MacGregor Smith and Daskalaki, 1988, Malakooti, 1994, Boysen et al., 2008].

- Le *coût de changements de séries*. Pour les lignes multi-produits, lorsque le produit d'un certain type est lancé, les stations qui doivent le traiter sont préparées à la fabrication de ce produit. La préparation consiste en plusieurs demarches, comme il a été mentionné dans le chapitre 1 : le chargement d'une pièce de produit, le positionnement du produit et des ressources/outils, le réglage des machines, le déchargement, le nettoyage de la station, etc. La plupart des travaux sur les systèmes d'ateliers à cheminement unique traitant des coûts de changements de séries sont consacrés aux problèmes d'ordonnancement de produits en ayant une configuration de ligne donnée. Nous rappelons que l'ordonnancement de produits répresente la dernière étape de la configuration préliminaire. Une recherche bibliographique détaillée sur ces problèmes et les méthodes de leur résolution a été réalisée par [Allahverdi et al., 2008]. Par ailleurs, les travaux concernant l'équilibrage de lignes de fabrication avec des coûts de changements de séries ne sont pas nombreux, nous ne pouvons citer que [Chakravarty and Shtub, 1986, Atmani, 1995, Yoosefelahi et al., 2012].

2.3.5 Contraintes technologiques

Il y a beaucoup de contraintes qui limitent le nombre de solutions admissibles d'un problème d'équilibrage de lignes de fabrication. Ces contraintes sont définies au stade qui précède la configuration, c'est-à-dire lors de la planification de processus (choix de gammes). En ayant l'ensemble des contraintes technologiques, le concepteur de ligne peut formuler un modèle du problème d'équilibrage.

Les contraintes qui sont utilisées dans la grande majorité des formulations sont les *contraintes de précédence*. Effectivement, les contraintes de précédence entre les opérations sont déjà présentes dans les problèmes de base de type SALBP [Salveson, 1955, Baybars, 1986]. Quant aux nombreuses généralisations du SALBP, les contraintes de précédence sont également prises en compte. L'un des critères de la classication des problèmes d'équilibrage des lignes de fabrication proposée par [Boysen et al., 2007] utilise les caractéristiques de graphes de précédence. Chaque graphe correspond à son propre type de produits. Si une ligne est dédiée, il y a un seul graphe

de précédence. Dans le cas des lignes multi-produits plusieurs graphes de précédence doivent être considérés. L'exigence principale pour les graphes de précédence réside dans leur acyclicité.

La *contrainte imposée sur le temps de cycle* joue un rôle important dans l'équilibrage des lignes de fabrication. En retournant au Tableau 2.1, nous voyons que soit le temps de cycle est donné soit il doit être minimisé. Dans les problèmes où le temps de cycle est donné, les temps de station ne doivent pas dépasser le temps de cycle, voir par exemple [Essafi et al., 2010, Scholl et al., 2010].

On peut distinguer deux types de *contraintes de capacité* utilisés lors de la modélisation d'un problème d'équilibrage de ligne : une *limite sur le nombre de ressources* affectées à une station [Aĝpak and Gökçen, 2005, Miralles et al., 2007] et une *borne sur le nombre de stations* [Belmokhtar et al., 2006, Dolgui and Ihnatsenka, 2009a]. Les contraintes de capacité reflètent la nécessité de respecter les dimensions des stations et de la ligne elle-même.

Nous pouvons aussi évoquer les contraintes liées à l'affectation des opérations aux stations. Nous allons les utiliser dans les chapitres suivants.

Les *contraintes d'inclusion* forcent certaines opérations à être affectées à la même station [Dolgui et al., 1999, Boysen and Fliedner, 2008, Simaria and Vilarinho, 2009, Delorme et al., 2012]. Il y a des situations industrielles pour lesquelles la précision ou la qualité de certaines opérations peut être perdue, si elles sont affectées à des stations différentes. Cela oblige à regrouper ces opérations sur les mêmes stations.

Les *contraintes d'exclusion* ont une nature opposée. Elles servent à représenter l'interdiction pour certaines ressources d'être affectées à la même station [Dolgui et al., 1999, Guschinskaya and Dolgui, 2009, Özcan and Toklu, 2009a, Scholl et al., 2010]. Elles reflètent soit une incompatibilité de ces ressources, soit un coût très élevé dans le cas où elles seraient affectées à la même station.

Les *contraintes sur un mode d'exécution des opérations* décrivent l'ordre d'exécution d'opérations sur les stations. Ainsi, les opérations peuvent être exécutées successivement [Lucertini et al., 1998, Özcan, 2010], en parallèle [Dolgui et al., 1999, Falkenauer, 2005, Belmokhtar et al., 2006] ou dans un mode mixte [Belmokhtar et al., 2007, Dolgui and Ihnatsenka, 2009a]. Un mode d'exécution séquentielle des opérations est inhé-

rent aux problèmes de type SALBP. Leurs généralisations gardent souvent cette hypothèse [Boysen et al., 2007]. L'exécution des opérations en parallèle, souvent observée dans l'industrie, sert à réduire le temps de cycle des lignes de fabrication et ainsi augmenter leur productivité. Par exemple, les modes parallèle et mixte sont souvent utilisés dans le domaine de l'usinage où les ressources correspondent à des boîtiers multi-broches effectuant simultanément des opérations d'usinage sur des pièces [Dolgui et al., 2006a, Guschinskaya et al., 2009].

2.3.6 Critères d'optimisation

Un critère d'optimisation implique une fonction objectif qui doit être soit minimisée soit maximisée. En retournant au SALBP, on peut distinguer deux critères de base qui sont souvent utilisés : la *minimisation du nombre de stations* [Salveson, 1955, Scholl and Klein, 1999, Kilincci, 2011] et la *minimisation du temps de cycle* [Klein and Scholl, 1996, Nearchou, 2007]. Il y a également les travaux sur les problèmes d'équilibrage qui ont pour but de *maximiser l'efficacité de ligne* par la minimisation du produit de deux critères de base, mais ils ne sont pas nombreux, nous ne pouvons citer que [Scholl, 1999, Wei and Chao, 2011].

Les critères de base concernent la capacité (minimisation du nombre de stations) et la productivité (minimisation du temps de cycle) d'une ligne de fabrication. Dans la plupart des cas, les problèmes avec ces critères n'utilisent pas les coûts. Néanmoins, la configuration d'une ligne de coût minimal est très importante pour le client. Afin de trouver une telle configuration, un **modèle orienté coût** doit être construit, en considérant les coûts d'installation qui sont fixes et les coûts variables. C'est une direction de recherche très importante, qui attire l'attention de beaucoup de chercheurs [Chakravarty and Shtub, 1986, Amen, 2000, Amen, 2001, Scholl and Becker, 2005, Dolgui and Ihnatsenka, 2009a, Essafi et al., 2010]. Un problème de choix d'équipement est souvent résolu simultanément avec un problème d'équilibrage. Dans ce cas, le critère de coût qui consiste à minimiser le coût total des équipements pour un temps de cycle donné est utilisé, voir par exemple [Pinnoi and Wilhelm, 1998, Bukchin and Tzur, 2000, Dolgui and Ihnatsenka, 2009b, Ege et al., 2009, Delorme et al., 2012]. De manière assez proche, certains modèles ont comme critère la *maximisation du bénéfice* [Boysen and Fliedner, 2008].

Il y a un critère lié à l'optimisation des temps de station : le critère de « *smoothness* » (le terme

anglais) qui signifie le niveau de régularité (lissage) des temps de station [Cakir et al., 2011, Mozdgir et al., 2013]. Cela implique soit la minimisation de l'écart entre le temps de chaque station et le temps de station moyen (le cas des lignes dédiées), soit la minimisation de l'écart entre les temps de la même station (le cas des lignes multi-produits).

Dans cette thèse nous nous intéresserons notamment à des modèles orientés coût.

2.4 Méthodes de résolution pour les problèmes de configuration de lignes de fabrication

La résolution des problèmes de configuration de lignes de fabrication nécessite l'utilisation de méthodes d'optimisation combinatoire. Il s'agit de trouver la meilleure solution selon les critères utilisés parmi les solutions admissibles, définies par des contraintes du problème. On distingue deux classes de méthodes d'optimisation combinatoire : les *méthodes exactes* et les *méthodes approchées*.

Les méthodes exactes garantissent l'optimalité des solutions obtenues. Il y a des logiciels d'optimisation, souvent appelés solveurs, qui mettent en œuvre des méthodes exactes afin de trouver les solutions optimales pour des problèmes d'optimisation combinatoire, y compris les problèmes de configuration de lignes de fabrication. Nous pouvons mentionner les solveurs les plus connus : IBM ILOG Cplex, GAMS, XPRESS, LINGO, etc. Ces solveurs peuvent être considérés comme des « boîtes noires ». On écrit un modèle mathématique du problème selon les exigences du langage de modélisation et on appele ensuite le solveur. Celui-ci renvoie alors les solutions trouvées ainsi que leurs caractéristiques [Belmokhtar et al., 2006, Corominas et al., 2008, Essafi et al., 2010, Battaïa and Dolgui, 2012]. Par exemple, le langage de modélisation AMPL est utilisée pour appeler Cplex. Par ailleurs, les langages de programmation généralistes comme C++ ou Java peuvent aussi être utilisés avec ces solveurs.

Quant aux méthodes exactes dédiées à la résolution des problèmes de configuration de lignes de fabrication, nous pouvons évoquer les suivantes :

- *Procédures par Séparation et Évaluation (PSE)* comme FABLE [Johnson, 1988], EUREKA

[Hoffmann, 1992], SALOME [Scholl and Klein, 1997], ABSALOM [Scholl et al., 2010]. Notons aussi que ce type de méthode est integré dans la plupart des solveurs.

- *Programmation par contraintes* [Belmokhtar, 2006, Topaloglu et al., 2012].
- *Programmation dynamique* [Jackson, 1956, Held et al., 1963, Easton, 1990, Nicosia et al., 2002].

Vu que le temps de calcul des méthodes exactes peut être trop grand à cause de la taille et de la complexité du problème étudié, le recours aux méthodes approchées represente parfois un meilleur choix. Ces méthodes visent à trouver des solutions de bonne qualité qui ne sont pas forcément optimales. La qualité de solutions obtenues est estimée en utilisant des bornes inférieures (si la fonction objectif doit être minimisée) ou supérieures (si on maximise la fonction objectif). Les bornes sont calculées afin d'estimer une erreur relative (gap) de la manière suivante :

$$\frac{HEU - LB}{HEU} \times 100\%,$$

où HEU - la valeur de solution obtenue par une méthode approchée, LB - la borne inférieure (supposons que la fonction objectif doit être minimisée). Il est à noter que les méthodes exactes incorporées dans les solveurs peuvent aussi donner une solution approchée, si le temps de calcul défini par l'utilisateur est épuisé avant qu'on arrive à une solution optimale. Dans ce cas, la meilleure solution admissible est proposée et l'erreur relative est indiquée. Elle est comparée à une borne inférieure pour estimer une erreur relative.

On subdivise souvent les méthodes approchées en deux parties : les heuristiques et les métaheuristiques. La différence entre ces deux réside dans leur complexité. Les métaheuristiques sont plus complexes que les heuristiques. Elles peuvent éviter les optima locaux, tandis que les heuristiques y sont très souvent bloquées. Les métaheuristiques ont une nature multifonctionnelle car elles sont adaptables à différents problèmes d'optimisation.

Les heuristiques sont basées sur les règles intuitives de choix de meilleur candidat (solution intermédiaire) lors du processus de résolution. Par exemple, les heuristiques gloutonnes choisissent un candidat qui contribue le mieux à la valeur de fonction objectif. Elles trouvent vite une solution, mais la qualité de cette solution peut être loin de l'optimum. Les heuristiques sont souvent utilisées pour trouver des solutions de départ. Ces solutions aident à initialiser

la recherche de la solution optimale de manière plus efficace. En outre, les heuristiques sont souvent employées afin d'améliorer le processus de résolution pour d'autres méthodes. Nous pouvons citer quelques travaux utilisant des heuristiques pour des problèmes de configuration de lignes de fabrication : [Talbot et al., 1986, Boctor, 1995, Dolgui et al., 2005a, Dimitriadis, 2006, Delorme et al., 2012].

Tandis que les heuristiques jouent plutôt un rôle secondaire dans la résolution des problèmes d'optimisation combinatoire, y compris pour les problèmes d'équilibrage de lignes de fabrication, les métaheuristiques sont souvent employées pour trouver les solutions de très bonne qualité. Les métaheuristiques sont caractérisées par une structure qui comporte un nombre d'étapes plus élevé par rapport aux heuristiques. Afin de résoudre les problèmes de configuration de lignes de fabrication, les métaheuristiques suivantes sont souvent utilisées :

- *Recuit simulé* [McMullen and Frazier, 1998, Vilarinho and Simaria, 2002, Özcan, 2010, Cakir et al., 2011].
- *Recherche tabou* [Chiang, 1998, Pastor et al., 2002, Lapierre et al., 2006, Özcan and Toklu, 2009c].
- *Algorithmes génétiques* [Rubinovitz and Levitin, 1995, Hwang et al., 2008, Kim et al., 2009, Hamzadayi and Yildiz, 2012].
- *Algorithme de colonies de fourmis* [Simaria and Vilarinho, 2009, Sabuncuoglu et al., 2009, Chica et al., 2011, Yagmahan, 2011].

2.5 Conclusion

Ce chapitre présente un état de l'art sur les problèmes de configuration de lignes de fabrication. Dans la plupart des travaux on suppose que l'ensemble des ressources est déjà défini. Donc, il ne reste qu'affecter les ressources données (on dit souvent qu'on affecte les opérations qui leur correspondent) à des stations. Le processus d'affectation des ressources à des stations est appelé l'équilibrage d'une ligne de fabrication. Pourtant, si les opérations peuvent être exécutées par plusieurs ressources, par exemple en utilisant différents équipements disponibles sur le marché, alors le problème de configuration d'une ligne comporte aussi un problème de choix de ressources. Le manque de recherche sur ce problème nous a motivé pour l'étudier. Dans le

chapitre 3 un tel problème qui consiste à choisir simultanément des ressources et une affectation des opérations aux stations sera présenté.

Afin de classer les nombreux problèmes d'équilibrage des lignes de fabrication, nous avons proposé de choisir les éléments communs, à savoir : la variabilité de ligne, les caractéristiques de ligne, les caractéristiques de temps, les caractéristiques de coût, les contraintes technologiques et les critères d'optimisation. Cette classification sera utilisée dans les chapitres suivants.

Nous avons décrit les problèmes de base de type SALBP. Les caractéristiques de ces problèmes sont les suivantes : la ligne est dédiée ; il y a un seul cheminement de production, les stations sont alignées de manière sérielle l'une après l'autre ; la ligne est cadencée ; les temps opératoires sont déterministes ; il y a des contraintes de précédence entre les opérations et il n'y pas d'autres restrictions sur l'affectation d'opérations ; les critères sont de trois types : la minimisation du nombre de stations, la minimisation du temps de cycle et la maximisation de l'efficacité par la minimisation du produit (nombre de stations)$\times$(temps de cycle). Les problèmes de type SALBP sont toujours d'actualité, quoique la plupart des recherches sont consacrées à des problèmes plus larges qui ont des hypothèses plus complexes.

Nous avons présenté un état de l'art non-exhaustif, en nous concentrant sur les hypothèses les plus répandues. Certaines de ces hypothèses seront étudiées dans les chapitres suivants. Finalement, nous avons mentionné les méthodes de résolution des problèmes de configuration de lignes de fabrication les plus utilisées. Elles se divisent en méthodes exactes et approchées. Les méthodes exactes garantissent l'obtention d'une ou plusieurs solutions optimales. Cependant, leur temps de calcul est souvent trop important étant donné la complexité du problème étudié. Les méthodes approchées ne donnent pas forcément des solutions optimales, mais ont un temps de calcul beaucoup plus petit par rapport à celui des méthodes exactes.

Chapitre 3

Problème d'équilibrage et de choix d'équipement pour des lignes dédiées

3.1 Introduction

Dans le chapitre 2 nous avons mentionné que les publications consacrées aux problèmes liés simultanément au choix de ressources et à l'équilibrage ne sont pas nombreuses en dépit de l'intérêt de cette problèmatique pour l'industrie. Dans ce chapitre, nous allons nous intéresser à un problème d'équilibrage de lignes dédiées associé au choix d'équipements. Afin de décrire les hypothèses propres à ce problème, la classification à six éléments présentée dans le chapitre 2 sera utilisée. Nous proposons ensuite une approche originale de résolution basée sur une réduction à un problème de partition d'ensemble et un algorithme de la génération de contraintes.

3.2 Définition du problème

La **variabilité de ligne**. Une ligne doit être configurée pour la fabrication de masse de produits du même type. Ainsi, la *ligne considérée est dédiée.*

Les **caractéristiques de ligne**. Le *schéma logique de la ligne est linéaire* : un sous-ensemble des opérations nécessaires est executée sur la première station, ensuite un autre sous-ensemble des opérations est effectué sur la deuxième station, et ainsi de suite jusqu'à ce que chaque opération de l'ensemble N, $N = \{1, \ldots, n\}$ soit exécutée exactement une seule fois. Un équipement de transport comme un tapis roulant par exemple, est utilisé pour le transfert des pièces de

produit d'une station à une autre dans la même direction. Il n'y a pas de stations parallèles et donc le *cheminement de production est unique*.

La *ligne est cadencée*, c'est-à-dire les produits restent sur chaque station pendant un temps constant qui est égal au temps de cycle c.

Les **caractéristiques de temps**. Les *temps opératoires sont déterministes*. Chaque station peut être équipée avec plusieurs équipements, appelés les *blocs*. L'ensemble des blocs disponibles, B, est supposé être connu. Chaque bloc effectue un ensemble d'opérations. Les opérations de tous les blocs affectés à la même station sont exécutées simultanément. Ainsi, le temps d'une station est déterminé par le temps opératoire le plus long sur cette station. Le *temps de cycle* de la ligne, à son tour, est égal au temps opératoire le plus long parmi toutes les n opérations. Si une borne supérieure sur le temps de cycle est donnée, alors, sans perte de généralité, nous supposons que le temps opératoire le plus long ne dépasse pas cette borne.

Les **caractéristiques de coût**.

Un *coût* $q(b) > 0$ est associé à chaque bloc $b \in B$. Un *coût d'installation constant* $C > 0$ est associé à chaque station.

Les **contraintes technologiques**.

Des *contraintes d'exclusion* sont définies sur l'ensemble des blocs B. Elles sont représentées par un ensemble E de sous-ensembles $E' \subset B$ tel que tous les blocs de E', $E' \in E$, ne peuvent pas être affectés à la même station. Par contre, n'importe quel sous-ensemble propre de E' peut être affecté à la même station. Ces contraintes décrivent les situations où certains outils ne peuvent pas être activés sur la même station à cause de leurs caractéristiques, par exemple, les dimensions. Nous supposons, sans perte de généralité, qu'il n'y a pas de sous-ensembles dans E tels que l'un contiendrait l'autre.

Les contraintes d'inclusion et de précédence sont données sur l'ensemble des opérations N.

Les *contraintes d'inclusion* sont représentées par un ensemble I de sous-ensembles $I' \subset N$ tel que toutes les opérations de I', $I' \in I$, doivent être affectées à la même station. Nous supposons sans perte de généralité qu'il n'y a pas d'intersection entre les sous-ensembles dans I.

Si une opération i précède une opération j, ce qui est désigné $i \rightarrow j$, alors j ne peut être exécutée ni sur la station de i, ni sur aucune des stations précédentes. Les *contraintes de*

précédence refletent des exigences technologiques pour les opérations. Ces relations sont transitives et irréflexives. Nous supposons qu'elles sont représentées par un graphe orienté acyclique $G_o = (N, A_o)$ dans lequel il y a un arc $(i, j) \in A_o$ si et seulement si une opération i précède une opération j. Selon cette définition, le graphe G_o coïncide avec sa fermeture transitive.

Notons que les relations d'exclusion et d'inclusion restreignent le contenu des stations, mais elles n'influencent pas la séquence des stations de la ligne. Par contre, les relations de précédence influencent en même temps la formation des stations et leur séquence.

Soit $|\cdot|$ la cardinalité. Il y a une borne supérieure m_0 sur le nombre de stations et une borne supérieure n_0 sur le nombre de blocs affectés à la même station. Le nombre de stations est limité à cause de la dimension de la ligne. Quant au nombre de blocs par station, $n_0 \in \{2, 3\}$ est souvent vérifié dans les applications réelles. Normalement, au plus un bloc est horizontalement affecté de chaque côté du tapis roulant et un bloc peut parfois être installé au-dessus du produit. Ce type de configuration a déjà été décrit dans [Dolgui et al., 2005b, Lapierre et al., 2006, Özcan and Toklu, 2009b, Delorme et al., 2012].

Comme il a déjà été mentionné, le *mode d'exécution des opérations* sur les stations est parallèle. Cela augmente considérablement la productivité de la ligne.

Le **critère d'optimisation**. Le critère d'optimisation pour ce problème est la minimisation du coût total :

$$mC + \sum_{k=1}^{m} \sum_{b \in W_k} q(b). \tag{3.1}$$

Le problème consiste à déterminer le nombre de stations, m, les ensembles de blocs affectés à ces stations, $W_1, \ldots, W_m$, et la séquence des stations telle que $|W_k| \leq n_0$, $k = 1, \ldots, m$, $m \leq m_0$, les opérations des blocs affectés constituent l'ensemble N, les contraintes d'exclusion, d'inclusion et de précédence sont vérifiées. Dans la suite, la notation W_r sera utilisée pour désigner une station.

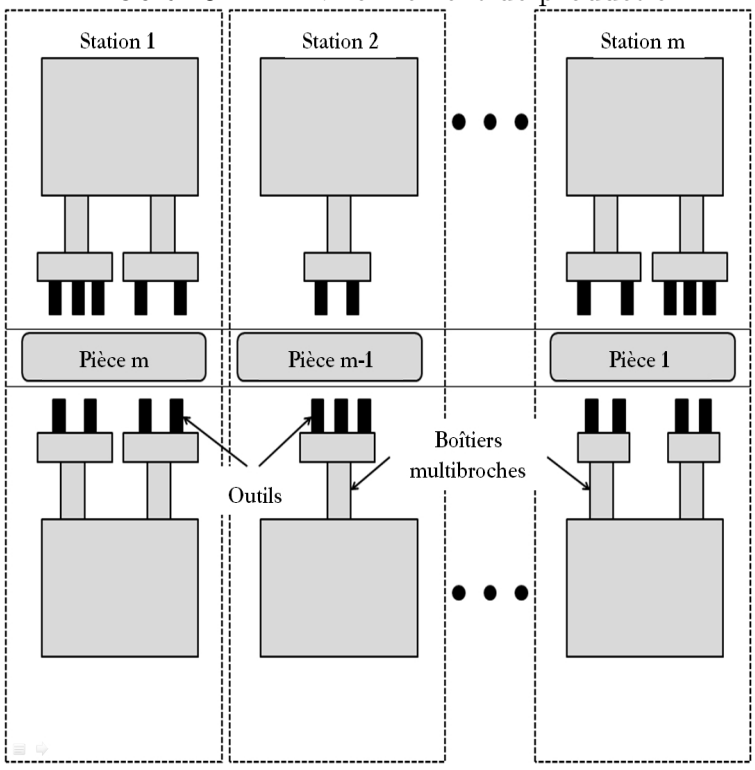
Nous appellons ce problème le problème P. Il a été formulé pour la première fois par Dolgui et al. [Dolgui et al., 2005b] et Belmokhtar et al. [Belmokhtar et al., 2006] qui ont souligné son importance pratique et ont montré la différence par rapport au SALBP et ses généralisations connues.

3.3 Origine du problème

Pour la première fois, le problème P a été rencontré dans les entreprises PCI-SCEMM (Saint-Étienne, France) et MZAL (Minsk, Biélorussie) qui conçoivent et produisent des lignes d'usinage automatisées sur commande de leurs clients. Le client pour lequel la ligne doit être conçue spécifie le produit et le volume de production annuel souhaité. Le constructeur de lignes conçoit le processus de production et définit la nomenclature des outils éligibles pour effectuer chaque opération (des blocs éligibles). Ensuite, il choisit des blocs et construit à partir de ces blocs une ligne de fabrication pour le client. L'optimisation pour la configuration de lignes est crucial pour ces entreprises.

De telles lignes sont souvent rencontrées dans l'usinage [Belmokhtar et al., 2006, Dolgui and Ihnatsenka, 2009a, Guschinskaya et al., 2009]. Sur chaque station une pièce à usiner est positionnée et plusieurs boîtiers multibroches (blocs) accèdent simultanément aux différents côtés de la pièce, voir une esquisse d'un tel environnement de production dans la Figure 3.1. Un exemple d'un boîtier multibroche est présenté dans la Figure 3.2.

FIGURE 3.1 – Environnement de production



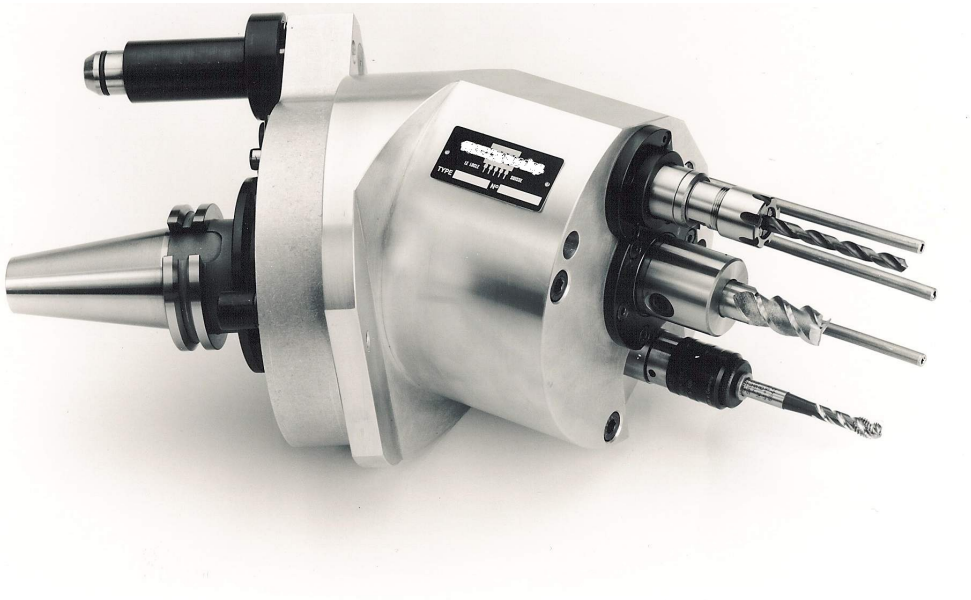


FIGURE 3.2 – Boîtier multibroche

Plusieurs outils sont regroupés dans le même bloc afin d'exécuter les opérations associées au même côté d'une pièce à usiner. Les outils du même bloc sont activés simultanément mais ils peuvent avoir des vitesses différentes. Notons également que si un bloc est choisi, alors toutes ces opérations doivent être accomplies parce que c'est économiquement inefficace de créer un bloc dont une partie ne sera pas utilisée.

Le constructeur de la ligne choisit des blocs et les affecte aux stations de sorte que l'ensemble des opérations des blocs sélectionnés donne l'ensemble des opérations nécessaires, le coût total soit minimisé, et les contraintes technologiques soient vérifiées. Il peut arriver que le problème P n'ait pas de solutions. Dans ce cas, l'ensemble des blocs disponibles doit être étendu et le problème P peut être re-résolu. Sachant qu'un bloc de capacité unitaire peut être conçu pour chaque opération on peut toujours trouver un ensemble des blocs qui permet de résoudre le problème P.

Dolgui et al. [Dolgui et al., 2005b] et Belmokhtar et al. [Belmokhtar et al., 2006] ont suggéré deux programmes en nombres entiers pour le problème P et ont décrit des expérimentations numériques effectuées en utilisant le solveur IBM ILOG Cplex. Dolgui et al. [Dolgui et al., 2006b] ont présenté une réduction du problème P au problème NP-difficile du chemin optimal sous contraintes, et Delorme et al. [Delorme et al., 2012] - au problème de recherche d'une clique de poids maximal. Dol-

gui et al. [Dolgui et al., 2008], Guschinskaya et Dolgui [Guschinskaya and Dolgui, 2009], et Dolgui et Ihnatsenka [Dolgui and Ihnatsenka, 2009a, Dolgui and Ihnatsenka, 2009b] ont aussi étudié une variante de ce problème dans laquelle les opérations des différents blocs sont exécutées séquentiellement ou dans un ordre mixte (parallèle et séquentiel).

3.4 Réduction du problème P au problème de partition d'ensemble

Afin de résoudre le problème P, nous proposons de le réduire au problème de partition d'ensemble. Le choix de cette méthode est justifié par une particularité des problèmes industriels liée au fait que le nombre de blocs sur la même station n'excède jamais trois. L'idée est alors venue d'énumérer tous les blocs localement admissibles sur une station et de n'inclure dans le modèle mathématique que des combinaisons de ces blocs. Cela élimine les contraintes d'inclusion et d'exclusion ainsi que les stations avec un nombre excessif des blocs. Le nombre total de ces combinaisons est un polynôme du nombre des blocs disponibles, $|B|$. Le degré de ce polynôme est le nombre maximal de blocs pouvant être installées sur la même station.

Nous allons maintenant introduire quelques notions qui seront utilisées par la suite. Nous appelons un ensemble de blocs $W \subset B$ une *station localement admissible* (LA-station) si $|W| \leq n_0$, chaque opération de W appartient à exactement un bloc de W et les contraintes d'exclusion, d'inclusion et de précédence sont vérifées pour les blocs et les opérations de W. Concernant les relations d'inclusion, cela signifie que soit aucune opération d'un ensemble $I' \in I$ n'appartient à un bloc de W, soit toutes les opérations d'un tel ensemble appartiennent aux blocs de W. Quant aux relations d'exclusion, une LA-station W ne doit pas contenir tous les blocs de n'importe quel ensemble $E' \in E$. Concernant les relations de précédence, aucune paire d'opérations i et j telle que $i \to j$, ne peut être affectée à la même LA-station W.

Nous appelons une séquence ordonnée $L = (W_1, \ldots, W_m)$ de LA-stations une *ligne admissible*, si $m \leq m_0$, chaque opération de l'ensemble N apparteint exactement à un bloc et les contraintes de précédence sont satisfaites pour toutes les n opérations. Notons que les contraintes d'exclusion et d'inclusion sont vérifiées pour une ligne admissible par définition car elle se compose seulement des LA-stations.

Désignons $\mathcal{W}$ et $\mathcal{L}$ l'ensemble de toutes les LA-stations et l'ensemble de toutes les lignes

admissibles, respectivement. Il est facile de voir qu'une solution optimale du problème P est une ligne admissible $L \in \mathcal{L}$ avec le coût total minimal qui est défini par (3.1).

On peut construire l'ensemble des LA-stations $\mathcal{W}$ de la manière suivante. Numérotons les blocs de l'ensemble B : $B_1, \ldots, B_T$, où $T = |B|$. Nous pouvons identifier chaque LA-station $W \in \mathcal{W}$ avec un k-uplet $(z_1, \ldots, z_k)$ des indices de ses blocs ou avec un 0-1 vecteur $(y_1, \ldots, y_T)$ tel que

$$y_i = \begin{cases} 1, & \text{si } B_i \in W, \\ 0, & \text{si } B_i \notin W \end{cases}.$$

En effet, $\mathcal{W}$ est un sous-ensemble de tous les k-uplets $(z_1, \ldots, z_k)$ tel que $1 \leq k \leq n_0$, et tous les z_i, $i = 1, \ldots, n_0$, sont des nombres distincts de l'ensemble $\{1, \ldots, T\}$, ou autrement, c'est un sous-ensemble de tous les vecteurs 0-1 T-dimensionnels avec au plus n_0 valeurs égales à un. Donc, $|\mathcal{W}| \leq \sum_{k=1}^{n_0} T^k = \frac{T^{n_0+1}-T}{T-1} = O(T^{n_0})$ et $|\mathcal{W}| \leq O(2^T)$, d'où $|\mathcal{W}| \leq O(\min\{T^{n_0}, 2^T\})$. L'ensemble $\mathcal{W}$ peut être contruit en temps $O(Poly(n, T, |E|) \min\{T^{n_0}, 2^T\})$, en considérant chaque k-uplet ou chaque 0-1 vecteur mentionnés plus haut et en vérifiant que la station correspondante est localement admissible. Ici, $Poly(n, T, |E|)$ est un polynôme de n, T et $|E|$. $|I|$ pourrait être un autre terme de ce polynôme, mais il n'y a pas d'intersection entre les ensembles dans I et donc $|I| \leq n$.

On peut construire une ligne admissible $L \in \mathcal{L}$ avec un coût total minimal de la façon suivante. Soit $K = |\mathcal{W}|$. Calculons le coût c_r de chaque LA-station W_r :

$$c_r = C + \sum_{b \in W_r} q(b), \quad r = 1, \ldots, K.$$

Introduisons les 0-1 variables x_r, $r = 1, \ldots, K$:

$$x_r = \begin{cases} 1, & \text{si LA-station } W_r \text{ est sélectionnée pour la ligne,} \\ 0, & \text{sinon.} \end{cases}$$

Ainsi, chaque 0-1 vecteur $x = (x_1, \ldots, x_K)$ correspond à une *ligne non-ordonnée* $LNO(x) = \{W_{r_1}, \ldots, W_{r_m}\}$ telle que $\{r_1, \ldots, r_m\} = \{t \mid x_t = 1,\ t = 1, \ldots, K\}$.

Identifions l'ensemble S_j des LA-stations qui contiennent une opération j : $S_j = \{W_r \mid j \in W_r,\ r = 1, \ldots, K\}$ pour $j = 1, \ldots, n$.

Soit $Oper(W_r)$ l'ensemble des opérations d'une LA-station W_r. Introduisons le graphe orienté $G_w = (\mathcal{W}, A_w)$, dans lequel l'ensemble des LA-stations est un ensemble des sommets, et il y a un arc $(W_p, W_r) \in A_w$ si et seulement si W_p et W_r n'ont pas d'opération commune et il y a des opérations $i \in Oper(W_p)$ et $j \in Oper(W_r)$ telles que $i \to j$. Soit $\mathcal{Y} = \{Y_1, \ldots, Y_a\}$ l'ensemble de tous les cycles dans G_w. Considérons la formulation de partition d'ensemble suivante, denotée comme le problème MinSPP.

Problème MinSPP

$$\min \sum_{r=1}^{K} c_r x_r, \tag{3.2}$$

sous les contraintes

$$\sum_{W_r \in S_j} x_r = 1, \quad j = 1, \ldots, n, \tag{3.3}$$

$$\sum_{W_r \in Y} x_r \leq |Y| - 1, \quad Y \in \mathcal{Y}, \tag{3.4}$$

$$1 \leq \sum_{r=1}^{K} x_r \leq m_0, \tag{3.5}$$

$$x_r \in \{0, 1\}, \quad r = 1, \ldots, K. \tag{3.6}$$

Les contraintes de « partition d'ensemble » (3.3) garantissent que les opérations des LA-stations choisies constituent l'ensemble N. Les contraintes de « cycle » (3.4) garantissent que le sous-graphe du graphe G_w, dont les sommets sont les LA-stations sélectionnées, est acyclique. Cela signifie que les stations choisies peuvent être ordonnées de sorte que les contraintes de précédence soient vérifiées. Les contraintes (3.5) garantissent que le nombre de LA-stations sélectionnées est au moins égal à un et au plus égal à m_0.

Proposition 1 *Le vecteur x vérifie les contraintes (3.3)-(3.6) si et seulement si les LA-stations de la ligne non-ordonnée correspondante $LNO(x)$ peuvent être ordonnées afin de former une ligne admissible.*

Preuve: La partie « si » est triviale. Considérons la partie « seulement si », c'est-à-dire, supposons que x vérifie (3.3)-(3.6). Soit la ligne non-ordonnée correspondante $LNO(x) = \{W_{r_1}, \ldots, W_{r_m}\}$. Vu que le sous-graphe du graphe G_w, dont les sommets sont $W_{r_1}, \ldots, W_{r_m}$, est

acyclique en raison de (3.4), ces sommets peuvent toujours être topologiquement ordonnés. Afin de simplifier la notation, soit $L(x) = (W_{r_1}, \ldots, W_{r_m})$ une séquence arbitraire de ces sommets qui est admissible au regard de G_w. La séquence $L(x)$ est donc une ligne admissible. En effet, les contraintes d'inclusion et d'exclusion sont satisfaites parce que $W_{r_1}, \ldots, W_{r_m}$ sont des LA-stations. Par ailleurs, les opérations des LA-stations sélectionnées constituent l'ensemble N en raison des contraintes (3.3). Les contraintes de précédence sont respectées parce que pour toute paire d'opérations i et j telle que $i \to j$ et $i \in Oper(W_{r_h})$, $j \in Oper(W_{r_g})$, nous avons $h \neq g$ en raison de la définition d'une LA-station, et $h < g$ en raison de la faisabilité topologique de la séquence $L(x)$. ∎

Nous en déduisons qu'une procédure en deux étapes peut être utilisée afin de résoudre le problème P. En premier lieu, il faut trouver une solution optimale, x^*, du problème MinSPP, et déterminer un ensemble optimal des LA-stations, $\{W_{r_1^*}, \ldots, W_{r_m^*}\}$, tel que $\{r_1^*, \ldots, r_m^*\} = \{t \mid x_t^* = 1,\ t = 1, \ldots, K\}$. En deuxième lieu, il faut trouver une séquence optimale, L^*, de ces stations, ce qui représente une séquence arbitraire admissible tenant compte du graphe G_w.

3.5 Prétraitement des données

Souvent, une information spécifique concernant un cas particulier peut être exploitée pour améliorer le calcul en réduisant la taille du problème pour ce cas. Dans l'article de Belmokhtar et al. [Belmokhtar et al., 2006] certaines procédures de prétraitement de données ont été développées avant l'utilisation d'un modèle de programmation linéaire en variables mixtes du problème P.

Nous allons proposer une autre procédure adaptée à notre approche. Ses objectifs sont les suivants :

- Réduire l'ensemble des blocs.
- Diminuer l'ensemble des LA-stations.
- Déterminer les LA-stations présentes dans la solution optimale (les LA-stations certaines).

Extension des relations de précédence. L'instrument principal du prétraitement est l'analyse des relations de précédence. Si l'ensemble des arcs A_o du graphe de précédence $G_o = (N, A_o)$ est grand, alors il y a une possibilité de réduire l'ensemble des blocs B et l'ensemble

des LA-stations $\mathcal{W}$ sans perte de la solution optimale.

L'ensemble A_o peut être élargi de la façon suivante. Si une opération i précède une opération j, c'est-à-dire, $(i, j) \in A_o$, et si $k \in I'$ alors l'arc (k, j) est ajouté à A_o. Après l'ajout de tous ces arcs, des opportunités d'ajouter des arcs transitifs dans le graphe G_o peuvent apparaître. Les arcs transitifs sont également ajoutés.

Réduction de l'ensemble des blocs B. Définissons les relations de précédence sur l'ensemble des blocs de sorte qu'un bloc b_1 précède un bloc b_2 . C'est exprimé comme $b_1 \Rightarrow b_2$, s'il y a des opérations $i \in b_1$, $j \in b_2$ telles que $i \rightarrow j$. Ces relations ne sont pas transitives et elles admettent des cycles. Introduisons un graphe non-orienté $G_b = (B, E_b)$ représentant des *conflits de blocs*, pour lequel l'ensemble des sommets coïncide avec l'ensemble des blocs B. Il y a une *arête de conflit* $\{b_1, b_2\} \in E_b$ dans ce graphe si et seulement si les blocs b_1 et b_2 ont soit une opération commune soit ils précédent l'un l'autre : $b_1 \Rightarrow b_2$ et $b_2 \Rightarrow b_1$. Les blocs reliés par une arête de conflit sont appelés (mutuellement) *en conflit*. Il est évident que des blocs en conflit ne peuvent pas être simultanément présents dans une solution admissible du MinSPP.

Nous allons appliquer un prétraitement des données du problème pour élargir l'ensemble des arêtes de conflit. Considérons les blocs b_1 et b_2 tels que $\{b_1, b_2\} \notin E_b$. Considérons l'ensemble B sans tous les blocs en conflit avec b_1 ou b_2. Si cet ensemble ne couvre pas l'ensemble de toutes les opérations N, alors une arête $\{b_1, b_2\}$ est introduite, c'est-à-dire que b_1 et b_2 ne peuvent pas apparaître simultanément dans une solution admissible.

L'ensemble des blocs B peut être réduit de la façon suivante. Si l'ensemble N de toutes les opérations n'est pas couvert par un bloc b quelconque et tous les blocs qui ne sont pas en conflit avec lui, alors b ne peut pas être présent dans une solution admissible. Il est donc retiré de l'ensemble B. Un ensemble initial $\mathcal{W}$ des LA-stations sera donc construit en utilisant l'ensemble réduit B.

Réduction de l'ensemble $\mathcal{W}$ des LA-stations. Si une LA-station contient au moins deux blocs en conflit, alors elle est enlevée de l'ensemble $\mathcal{W}$. S'il y a deux LA-stations W_r et W_p telles que leurs ensembles d'opérations coïncident et $\sum_{b \in W_r} q(b) \leq \sum_{b \in W_p} q(b)$, alors la station W_p est retirée de $\mathcal{W}$. Pour les LA-stations restantes, nous effectuons les calculs suivants. Le graphe G_w, défini avant la formulation du problème MinSPP, est construit sur la base de l'ensemble

réduit $\mathcal{W}$.

Deux stations W_i et W_j sont appelées *en conflit* (mutuellement) si elles ont une opération commune ou il y a des arcs (W_i, W_j) et (W_j, W_i) dans le graphe G_w. Des stations en conflit ne peuvent pas être présentes dans une solution admissible du MinSPP. Si l'ensemble N de toutes les opérations n'est pas couvert par une station W quelconque et les stations qui ne sont pas en conflit avec elle, alors W ne peut pas être présente dans une solution admissible. Elle est donc enlevée de l'ensemble $\mathcal{W}$.

Notons que l'analyse complète des stations en conflit nécessite un temps $O(|\mathcal{W}|^2)$ qui peut être très grand pour des situations pratiques où $|\mathcal{W}|$ peut atteindre des centaines de milliers. Dans ce cas, une analyse réduite des conflits entre les stations et les blocs peut être effectuée. Appelons une station W et un bloc b *en conflit* si W est en conflit avec une station $W' = \{b\}$. Si l'ensemble N de toutes les opérations n'est pas couvert par une LA-station W quelconque et par les blocs qui ne sont pas en conflit avec W, alors W est retirée de l'ensemble $\mathcal{W}$. L'analyse réduite peut être mise en œuvre en temps $O(nT|\mathcal{W}|)$.

Les ensembles S_j des LA-stations qui contiennent une opération j, $j \in N$, sont construits en se fondant sur l'ensemble réduit $\mathcal{W}$. Ensuite, si $k \neq j$ et l'ensemble S_k comporte l'ensemble S_j comme un sous-ensemble propre, alors des LA-stations de l'ensemble $S_k \backslash S_j$ sont enlevées de $\mathcal{W}$.

LA-stations certaines. Si une opération j appartient à une seule LA-station W_r, c'est-à-dire, $S_j = \{W_r\}$, alors W_r est présente dans une solution optimale, et nous définissons $x_r = 1$.

Resserrement des inégalités (3.4) et (3.5). Soit Q une collection des ensembles non-intersectés S_j, y compris le cas d'un seul ensemble S_j. L'inégalité (3.5) peut être resserrée de sorte que les variables x_r correspondantes aux LA-stations de Q en sont retirées (mais pas des autres contraintes ni de la fonction objectif), la borne de côté gauche est remplacée par 0 et la borne de côté droit est ré-initialisée comme $m_0 := m_0 - |Q|$.

Une procédure similaire peut être appliquée aux inégalités de « cycle » (3.4) : si toutes les LA-stations de la collection Q définie plus haut appartiennent à un certain cycle Y, alors l'égalité correspondante (3.4) peut être resserrée de sorte que les variables x_r correspondantes aux LA-stations de Q en sont enlevées (là aussi, elles ne sont pas supprimées des autres contraintes ni

de la fonction objectif) et la borne de côté droit est ré-initialisée comme $|Y|-1 := |Y|-1-|Q|$.

Toutes les actions de prétraitement décrites plus haut sauf la dernière sont réitératives. Pour le resserrement d'une des inégalités (3.4) ou (3.5), seulement un ensemble Q (même si c'est un seul ensemble S_j) peut être utilisé.

Une application des actions de prétraitement « LA-stations certaines » et « Resserrement des inégalités (3.4) et (3.5) » peut conduire à ce que dans la formulation du MinSPP une égalité apparaît dans laquelle le côté droit contient un zéro. Dans ce cas, toutes les variables du côté gauche sont mises à zéro. Cela implique l'élimination des LA-stations correspondantes de l'ensemble $\mathcal{W}$.

3.6 Heuristiques gloutonnes

Dans le chapitre 2 nous avons mentionné que pour des problèmes combinatoires des heuristiques sont souvent mises en œuvre pour trouver des solutions de départ. Ensuite, ces solutions peuvent servir de base à une recherche plus complexe et plus efficace. Afin de résoudre le problème P, nous proposons d'utiliser deux types d'heuristiques gloutonnes avant de recourir à la méthode plus complexe. Ces heuristiques visent à trouver de bonnes solutions approchées du MinSPP.

La première heuristique, appelée **heuristique gloutonne**, sélectionne une LA-station avec le coût minimal par opération. L'algorithme est le suivant :

Heuristique gloutonne

1. Initialiser une ligne non-ordonnée $L = \phi$.
2. Tant que l'ensemble des LA-stations $\mathcal{W}$ n'est pas vide, exécuter les actions suivantes :
   (a) Retirer de $\mathcal{W}$ une LA-station W_r qui minimise $c_r/|Oper(W_r)|$.
   (b) Ajouter W_r à L.
   (c) Retirer de $\mathcal{W}$ toutes les stations qui ont des blocs en conflit ou constituent des cycles avec les stations dans L.
3. Si L ne couvre pas toutes les opérations, alors arrêter : aucune solution admissible n'est construite.

4. Ordonner topologiquement les stations dans L pour construire une ligne admissible et renvoyer L.

Il peut advenir qu'aucune solution admissible ne soit fournie par l'heuristique gloutonne. Afin de surmonter cette déficience, nous recourons à une **heuristique gloutonne randomisée**. Elle construit un sous-ensemble des LA-stations de manière itérative et aléatoire. Ensuite, une station avec le coût minimal par opération est prise dans ce sous-ensemble. La cardinalité de ce sous-ensemble, désignée comme s, est un paramètre. Dans notre cas, nous avons défini $s = \max\{2, 0.25|\mathcal{W}|\}$. L'algorithme est le suivant :

Heuristique gloutonne randomisée

1. Initialiser une ligne non-ordonnée $L = \phi$.
2. Tant que l'ensemble des LA-stations $\mathcal{W}$ n'est pas vide, exécuter les actions suivantes :
   (a) Sélectionner des LA-stations $W_1, \ldots, W_s$ de $\mathcal{W}$ aléatoirement.
   (b) Sélectionner une station W_r de $W_1, \ldots, W_s$ qui minimise $c_r/|Oper(W_r)|$.
   (c) Ajouter W_r à L.
   (d) Retirer de $\mathcal{W}$ toutes les stations qui ont des blocs en conflit ou constituent des cycles avec les stations dans L.
3. Si L ne couvre pas toutes les opérations, alors arrêter : aucune solution admissible n'est construite.
4. Ordonner topologiquement les stations dans L pour construire une ligne admissible et renvoyer L.

L'heuristique gloutonne randomisée est exécutée plusieurs fois pour augmenter la probabilité d'obtenir une solution admissible à petit coût.

3.7 Algorithme de génération de contraintes

Lorsqu'on résout le problème MinSPP, de graves difficultés surviennent si le nombre de LA-stations, $|\mathcal{W}|$, atteint des dizaines de milliers et le nombre de cycles, a, atteint plusieurs millions.

Alors, le problème n'entre pas dans la mémoire de l'ordinateur. Afin de faire face à des instances du MinSPP avec ces grandes dimensions, nous avons mis en œuvre un algorithme itératif de génération de contraintes, voir [Nemhauser and Wolsey, 1988]. L'algorithme commence avec un problème relaxé, désigné comme *Relax-MinSPP*, avec les contraintes de « cycle » omises. À chaque itération, le problème Relax-MinSPP est résolu par un solveur commercial et sa solution est analysée. Si la solution est inadmissible au regard du problème original MinSPP, alors les violations de contraintes identifiées sont utilisées pour générer des contraintes supplémentaires. Ensuite, elles sont incorporées dans un nouveau problème Relax-MinSPP qui est résolu à l'itération suivante. Le processus est répété jusqu'à ce qu'une solution admissible soit obtenue ou que l'infaisabilité soit prouvée.

La stratégie d'ajout des contraintes joue un rôle essentiel dans l'algorithme. Une possibilité consiste à ajouter à chaque itération des contraintes détruisant tous les cycles observées dans la solution courante. Pourtant, les expérimentations ont montré que cette approche génère trop de problèmes relaxés pour les problèmes MinSPP à grande dimension. Afin d'accélérer le processus de résolution, seulement certaines contraintes de « cycle » sont ajoutées. Notons que le nombre de ces contraintes doit être relativement faible pour maintenir une complexité de calcul raisonnable.

Notre statégie de sélection des contraintes se fonde sur une observation empirique. Celle-ci montre que les cas où une solution du problème Relax-MinSPP contient des cycles avec plus que deux sommets sont très rares. Donc, nous ajoutons des contraintes correspondant aux cycles avec deux sommets. Malgré cela, même les cycles avec deux sommets sont très nombreux dans les instances réalistes. Nous proposons donc d'utiliser un sous-ensemble de ces cycles. Nous le définissons de la façon suivante.

Nous appelons un 4-uplet de blocs (b_1, b_2, b'_1, b'_2) un *groupe de blocs en conflit*, s'ils n'ont aucune opération commune et que $b_1 \Rightarrow b'_1$ et $b'_2 \Rightarrow b_2$, y compris les cas $b_1 = b_2$ et/ou $b'_1 = b'_2$. Si une LA-station W contient b_1 et b_2, et une LA-station W' contient b'_1 et b'_2, alors W et W' sont en conflit. Toutes les deux ne peuvent pas être présentes dans une solution admissible du problème MinSPP. Donc, une contrainte de « cycle » peut être introduite pour elles. En outre, l'ensemble $\{W \mid b_1, b_2 \in W\} \cup \{W' \mid b'_1, b'_2 \in W'\}$ peut contenir au plus une LA-station dans

une solution admissible quelconque. En ayant (b_1, b_2, b'_1, b'_2), la contrainte correspondante peut être écrite comme $\sum_{W_r:\ b_1,b_2\in W_r} x_r + \sum_{W_p:\ b'_1,b'_2\in W_p} x_p \leq 1$. Cette contrainte est équivalent de l'ensemble de contraintes de « cycle » associées au groupe de blocs en conflit (b_1, b_2, b'_1, b'_2).

Soit $\mathcal{G}$ l'ensemble de tous les groupes de blocs en conflit. Rappelons que $\mathcal{Y}$ dans (3.4) signifie une famille de sous-ensembles Y de $\mathcal{W}$ qui représente tous les cycles dans le graphe G_w. Pour la description formelle de l'algorithme de génération de contraintes, nous désignons $\overline{\mathcal{Y}}$ un sous-ensemble des cycles, et $\overline{\mathcal{G}}$ un sous-ensemble des groupes de blocs en conflit. Ainsi, le problème Relax-MinSPP peut être écrit comme suit :

Problème Relax-MinSPP

$$\min \sum_{r=1}^{K} c_r x_r, \tag{3.7}$$

sous les contraintes

$$\sum_{r\in S_j} x_r = 1, \quad j = 1, \ldots, n, \tag{3.8}$$

$$\sum_{r\in Y} x_r \leq |Y| - 1, \quad Y \in \overline{\mathcal{Y}}, \tag{3.9}$$

$$\sum_{W_r:\ b_1,b_2\in W_r} x_r + \sum_{W_p:\ b'_1,b'_2\in W_p} x_p \leq 1, \quad (b_1, b_2, b'_1, b'_2) \in \overline{\mathcal{G}} \tag{3.10}$$

$$1 \leq \sum_{r=1}^{K} x_r \leq m_0, \tag{3.11}$$

$$x_r \in \{0, 1\}, \quad r = 1, \ldots, K. \tag{3.12}$$

Il est donc possible de donner une description formelle de l'algorithme de génération de contraintes.

Algorithme de génération de contraintes

1. Mettre $\overline{\mathcal{Y}} = \phi$ et $\overline{\mathcal{G}} = \phi$.
2. Si les limites de mémoire et de temps ne sont pas dépassées, alors résoudre le Relax-MinSPP. Sinon, arrêter.
3. Si le Relax-MinSPP est inadmissible, alors arrêter : le MinSPP est aussi inadmissible.

4. Mettre $L = \{W_r \mid x_r = 1\}$.
5. Si L ne contient aucune cycle, alors arrêter : L est une ligne non-ordonnée optimale.
6. Si L contient des cycles, alors trouver tous les groupes de blocs en conflit (b_1, b_2, b'_1, b'_2) dans l'ensemble de blocs dans L et les ajouter à $\overline{\mathcal{G}}$.
7. Si L contient des cycles avec au moins trois sommets, alors ajouter ces cycles à $\overline{\mathcal{Y}}$.
8. Retourner à l'étape 2.

Les résultats possibles de l'algorithme sont les suivantes :

- Une solution optimale acyclique est obtenue et elle correspond à une ligne optimale.
- L'infaisabilité du Relax-MinSPP et, donc, du MinSPP est prouvée.
- À la dernière itération, le Relax-MinSPP n'est pas résolu à l'optimalité, mais une solution admissible est trouvée et elle est acyclique. Alors, cette solution correspond à une ligne admissible avec une erreur relative renvoyée par le solveur.
- À la dernière itération, le Relax-MinSPP n'est pas résolu en raison des limites de mémoire ou de temps. Dans ce cas, seulement une borne inférieure du coût optimal est obtenue.

3.8 Exemple du problème P

Afin de clarifier le problème P et la méthode de résolution choisie, considérons un exemple dans lequel l'ensemble des opérations est $N = \{1, 2, 3, 4, 5, 6\}$ et l'ensemble des blocs disponibles est $B = \{b_1, \ldots, b_9\}$. Le coût d'installation de chaque station est $C = 10$. Les opérations et les coûts des blocs sont donnés dans le Tableau 3.1.

Tableau 3.1 – Blocs et leur caractéristiques pour l'exemple du problème P.

Bloc	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9
Opérations du bloc	$\{1,4\}$	$\{2\}$	$\{4\}$	$\{6\}$	$\{1,6\}$	$\{2,3\}$	$\{5\}$	$\{2,3,5\}$	$\{1,5,6\}$
Coût	3	1	2	2	3	3	2	5	5

Les contraintes d'exclusion sont données par $E = \Big\{\{b_1, b_7\}, \{b_2, b_7\}, \{b_3, b_7\}, \{b_4, b_7\}, \{b_5, b_7\}, \{b_6, b_7\}, \{b_3, b_4\}, \{b_4, b_6\}, \{b_4, b_8\}\Big\}$, les contraintes d'inclusions sont représentées par $I =$

$\Big\{\{2,3\}\Big\}$, les contraintes de précédence sont données par $1 \rightarrow 2$, $4 \rightarrow 5$, et les bornes supérieures sur le nombre de blocs de la même station et sur le nombre de stations sont $n_0 = 2$ et $m_0 = 3$, respectivement.

Le prétraitement fournit les résultats suivants. Les relations de précédence sont étendues par l'ajout de $1 \rightarrow 3$. Les paires de blocs en conflit sont : 1) $\{b_1, b_3\}$, $\{b_1, b_5\}$, $\{b_1, b_9\}$, $\{b_2, b_6\}$, $\{b_2, b_8\}$, $\{b_4, b_5\}$, $\{b_4, b_9\}$, $\{b_5, b_9\}$, $\{b_6, b_8\}$, $\{b_7, b_8\}$, $\{b_7, b_9\}$, et $\{b_8, b_9\}$ qui ont des opérations communes, et 2) $\{b_2, b_1\}$, $\{b_2, b_3\}$, $\{b_2, b_4\}$, $\{b_2, b_5\}$, $\{b_2, b_7\}$, $\{b_2, b_9\}$, $\{b_3, b_4\}$ qui, avec les blocs n'étant pas en conflit avec eux, ne couvrent pas l'ensemble N des opérations. En se fondant sur les conflits de blocs, le bloc b_2 est enlevé parce que l'opération 3 n'est pas couverte ni par ce bloc, ni par les blocs qui ne sont pas en conflit avec b_2.

L'ensemble des LA-stations générées est $\mathcal{W} = \{W_1, \ldots, W_{11}\}$. Leurs compositions et coûts sont montrés dans le Tableau 3.2.

Tableau 3.2 – Compositions et coûts des LA-stations pour l'exemple du problème P.

LA-stations	W_1	W_2	W_3	W_4	W_5	W_6	W_7	W_8	W_9	W_{10}	W_{11}
Bloc	b_1	b_3	b_4	b_5	b_6	b_8	b_9	b_1, b_4	b_3, b_5	b_3, b_6	b_7
Opérations	$1, 4$	4	6	$1, 6$	$2, 3$	$2, 3, 5$	$1, 5, 6$	$1, 4, 6$	$1, 4, 6$	$2, 3, 4$	5
Coût c_r	13	12	12	13	13	15	15	15	15	15	12

Aucune LA-station ne contient des blocs mutuellement en conflit. La station W_9 est enlevée de l'ensemble $\mathcal{W}$ parce que W_8 et W_9 contiennent le même ensemble des opérations et ont le même coût.

Le graphe $G_w = (\mathcal{W}, A_w)$ est construit. L'ensemble des arcs est A_w=$\{(W_1, W_5)$, (W_1, W_6), (W_1, W_{11}), (W_2, W_6), (W_2, W_7), (W_2, W_{11}), (W_4, W_5), (W_4, W_6), (W_4, W_{10}), (W_7, W_5), (W_7, W_{10}), (W_8, W_5), (W_8, W_6), (W_8, W_{11}), (W_{10}, W_7), $(W_{10}, W_{11})\}$.

Les stations en conflit sont $\{W_1, W_2\}$, $\{W_1, W_4\}$, $\{W_1, W_7\}$, $\{W_1, W_8\}$, $\{W_1, W_{10}\}$, $\{W_2, W_8\}$, $\{W_2, W_{10}\}$, $\{W_3, W_4\}$, $\{W_3, W_7, \}$, $\{W_3, W_8\}$, $\{W_4, W_7\}$, $\{W_4, W_8\}$, $\{W_5, W_6\}$, $\{W_5, W_{10}\}$, $\{W_6, W_7\}$, $\{W_6, W_{10}\}$, $\{W_6, W_{11}\}$, $\{W_7, W_8\}$, $\{W_7, W_{10}\}$, $\{W_7, W_{11}\}$, $\{W_8, W_{10}\}$. Aucune LA-station ne peut être enlevée en se fondant sur les conflits des stations.

Les ensembles S_j des LA-stations qui contiennent l'opération j sont donnés dans le Ta-

bleau 3.3, $j = 1, \ldots, 6$.

TABLEAU 3.3 – Ensembles S_j pour l'exemple du problème P.

S_1	S_2	S_3	S_4	S_5	S_6
W_1	W_5	W_5	W_1	W_6	W_3
W_4	W_6	W_6	W_2	W_7	W_4
W_7	W_{10}	W_{10}	W_8	W_{11}	W_7
W_8			W_{10}		W_8

Aucun ensemble S_k ne contient un autre ensemble S_j comme un sous-ensemble propre. Donc, aucune station n'est retirée de l'ensemble $\mathcal{W}$. Il n'y a pas de LA-stations définies ni d'ensembles non-intersectés S_j.

Le problème Relax-MinSPP initial dans cet exemple ne contient aucune contrainte de « cycle ». La fonction objectif $13x_1+12x_2+12x_3+13x_4+13x_5+15x_6+15x_7+15x_8+15x_{10}+12x_{11}$ doit être minimisée, sous les contraintes de « partition d'ensemble » $x_1 + x_4 + x_7 + x_8 = 1$, $x_5+x_6+x_{10} = 1$, $x_1+x_2+x_8+x_{10} = 1$, $x_6+x_7+x_{11} = 1$, $x_3+x_4+x_7+x_8 = 1$, les contraintes de « nombre des stations » $1 \leq x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 + x_{10} + x_{11} \leq 3$, et les contraintes 0-1 $x_i \in \{0, 1\}$, $i = 1, \ldots, 11$.

Supposons que le problème Relax-MinSPP soit résolu avec la solution suivante : $LNO = \{W_7, W_{10}\}$. Cette solution est cyclique : l'opération $1 \in W_7$ précède les opérations $(2, 3) \in W_{10}$, tandis que l'opération $4 \in W_{10}$ précède l'opération $5 \in W_7$. On peut trouver des groupes de blocs en conflit. Dans ce cas, un tel groupe est trouvé, c'est (b_3, b_6, b_9, b_9). Ajoutons-le à $\overline{\mathcal{G}}$. La contrainte correspondante (3.10) est ensuite incluse dans le Relax-MinSPP : $x_7 + x_{10} \leq 1$. À l'itération suivante la solution optimale est trouvée : $LNO = \{W_6, W_8\}$. Cette solution n'a pas de cycles. Donc, elle est optimale pour le problème original MinSPP. La séquence topologiquement admissible des stations W_6 et W_8 au regard du graphe G_w est $L^* = (W_8, W_6)$. C'est une solution optimale du problème P. Son coût total est égal à 30. La première station W_8 contient deux blocs b_1={1,4} et b_4={6}, et la deuxième station W_6 contient un bloc b_8={2,3,5}.

3.9 Résultats numériques

Nos expérimentations numériques pour le problème P se composent de quatre stades principaux :

1. Représenter les données sous la forme nécessaire.
2. Prétraiter les données.
3. Résoudre le problème MinSPP.
4. Convertir la solution du MinSPP en solution optimale du problème P.

Nous nous concentrons sur le stade de résolution du problème MinSPP.

La programmation linéaire en nombres entiers a été intensivement utilisée pour des problèmes de partition d'ensemble. Plusieurs méthodes de résolution sont devéloppées pour ces problèmes, y compris celles qui emploient des solveurs comme IBM ILOG Cplex, voir, par exemple, [Albareda-Sambola et al., 2011, Baldacci et al., 2011, Fanjul-Peyro and Ruiz, 2012]. Nous avons aussi utilisé le solveur IBM ILOG Cplex version 12.2. pour résoudre le problème Relax-MinSPP. Les expérimentations ont été effectuées sur un ordinateur avec le processeur AMD Phenom 2.8 Mhz et une mémoire à accès direct de 4 Go sous le système d'exploitation MS Windows XP (32 bit).

Dans nos expérimentations, la limite de temps pour chaque lancement de Cplex (pour résoudre le Relax-MinSPP) a été fixée à 600 secondes, et la limite de temps total pour résoudre chaque instance du MinSPP a été fixée à deux heures. L'algorithme de génération de contraintes a été mis en œuvre dans GAMS [gam, 2012]. Les autres parties algorithmiques ont été codées en C++.

Dans nos expérimentations deux familles de données aléatoirement générées ont été utilisées. Elles sont des extensions des données de référence de [Belmokhtar et al., 2006] et [Delorme et al., 2012] obtenues avec les mêmes générateurs de données.

La **famille 1** se compose de 3 groupes d'instances avec 75, 95 et 115 opérations. Chaque groupe est divisé en quatre séries (e.g., 75_1, 75_2, 75_3, et 75_4) avec une densité de contraintes de précédence faible, moyenne, élevée et très élevée, respectivement. La série « 75 » est prise de [Belmokhtar et al., 2006], où elle a été utilisée pour tester l'approche de programma-

tion linéaire en nombres entiers mixtes. Les autres séries ont été créées avec le même générateur et des paramètres identiques à [Belmokhtar et al., 2006], mais avec des valeurs plus larges de N, B et m_0. Chaque série se compose de 20 instances dont les caractéristiques sont données dans le Tableau 3.4. Dans ce tableau, B_{init} et B_{pret} sont les ensembles de blocs avant et après le prétraitement, respectivement. $\%Prec$ est la densité des contraintes de précédence après le prétraitement, définie comme $2|A_o|/(n^2 - n) \times 100\%$. Les valeurs de $|B_{init}|$, $|B_{pret}|$, $\%Prec$, $|I|$ et $|E|$ sont les moyennes pour 20 instances de chaque série. Le nombre maximal de blocs par station est toujours $n_0 = 4$. Le nombre moyen d'opérations dans un bloc est 2,3 dans toutes les séries sauf celles avec l'indice 4 pour lesquelles il est 1,8. Tous les ensembles d'exclusion $E' \in E$ comportent deux blocs, et les ensembles d'inclusion $I' \in I$ ont 2,2 opérations en moyenne.

La **famille 2** se compose de 4 groupes d'instances avec 70, 90, 110, et 130 opérations. Chaque groupe est divisé en trois séries (e.g., 70_1, 70_2, et 70_3) avec une densité de contraintes de précédence faible, moyenne et élevée, respectivement. Les autres paramètres sont similaires à ceux de la famille 1 : $n_0 = 4$, le nombre moyen d'opérations dans un bloc est égal à 2,3 dans les séries avec les indices 1 et 2. Pour les séries avec l'indice 3 il est égal à 1,8. Tous les ensembles d'exclusion $E' \in E$ comportent deux blocs, et les ensembles d'inclusion $I' \in I$ ont 2,2 opérations en moyenne. Le groupe d'instances « 70 » a été pris de [Delorme et al., 2012], où il a été utilisé pour tester l'approche de la clique de poids maximal. Les autres groupes ont été créés avec le même générateur que ceux de [Delorme et al., 2012]. Les caractéristiques numériques de la famille 2 sont données dans le Tableau 3.5.

Notons que les procédures de prétraitement, décrites dans la section 3.5, ont permis de réduire environ 1000 fois l'ensemble de LA-stations $\mathcal{W}$. Par exemple, le nombre de LA-stations pour les instances avec $n \in [75, 95]$, $|B| \in [200, 300]$, $n_0 = 4$ et $m_0 \in [34, 50]$, a été réduit de 10^8 à 10^5.

Pour résoudre chaque instance du problème P, nous utilisons la procédure à cinq étapes suivante.

Procédure de résolution pour le problème P :

1. Générer l'ensemble $\mathcal{W}$ des LA-stations.
2. Appliquer l'heuristique gloutonne.

TABLEAU 3.4 – Caractéristiques numériques de la famille 1

Séries	n	m_0	$\|B_{init}\|$	$\|B_{pret}\|$	$\%Prec$	$\|I\|$	$\|E\|$
75_1	75	34	215	206	0,27	10	1153
75_2	75	34	215	207	0,48	9	1150
75_3	75	34	215	207	0,63	8	1151
75_4	75	34	195	185	0,91	7	948
95_1	95	44	275	262	0,34	13	1894
95_2	95	44	275	269	0,57	13	1884
95_3	95	44	275	262	0,72	12	1881
95_4	95	44	257	241	0,92	10	1650
115_1	115	50	336	318	0,39	12	5395
115_2	115	50	335	311	0,60	12	5176
115_3	115	50	335	319	0,75	11	4920
115_4	115	50	310	299	0,93	9	2711

TABLEAU 3.5 – Caractéristiques numériques de la famille 2

Séries	n	m_0	$\|B_{init}\|$	$\|B_{pret}\|$	$\%Prec$	$\|I\|$	$\|E\|$
70_1	70	34	133	114	0,27	10	442
70_2	70	34	133	118	0,62	9	439
70_3	70	34	130	122	0,87	7	422
90_1	90	40	173	148	0,34	12	742
90_2	90	40	173	151	0,63	11	744
90_3	90	40	171	158	0,85	10	732
110_1	110	44	213	182	0,36	12	1130
110_2	110	44	213	184	0,66	10	1126
110_3	110	44	211	195	0,87	9	1111
130_1	130	46	253	218	0,40	11	1597
130_2	130	46	253	221	0,71	11	1598
130_3	130	46	251	225	0,88	9	1566

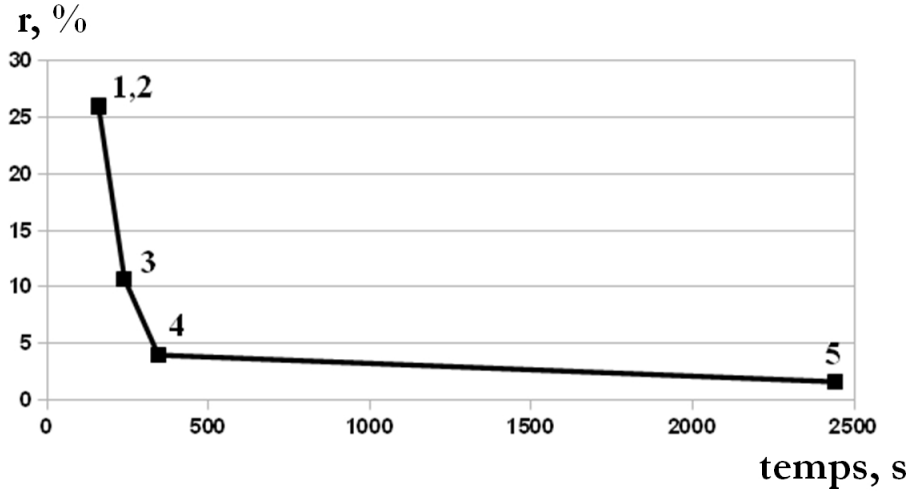
3. Exécuter l'heuristique gloutonne randomisée 1000 fois.
4. Appliquer l'algorithme de génération de contraintes pour l'ensemble des LA-stations $\mathcal{W}'$ extraites de toutes les solutions gloutonnes : $\mathcal{W}' = \bigcup L^{(k)}$, où chaque $L^{(k)}$ est une ligne obtenue après le k-ième lancement de l'algorithme de l'heuristique gloutonne randomisée, $k = 1, ..., 1000$.
5. Employer l'algorithme de génération de contraintes sur l'ensemble de toutes les LA-stations $\mathcal{W}$.

Ainsi, les étapes 2, 3 et 4 n'explorent qu'une partie de l'espace de solutions (méthodes approchées), tandis que l'étape 5 implique la recherche exhaustive de tout l'espace de solutions (méthode exacte) basé sur l'ensemble de toutes les LA-stations $\mathcal{W}$.

Pour les séries avec les indices 3 et 4, les étapes 3 et 4 de la procédure de résolution ont été omises parce qu'elles prennent plus de temps que la méthode exacte de l'étape 5.

La Figure 3.3 illustre la dynamique du processus de résolution pour la série 75_1. Chacun des quatre points est associé aux étapes de la procédure de résolution (les étapes 1 et 2, l'étape 3, l'étape 4, et l'étape 6, respectivement), l'erreur relative moyenne des solutions (l'axe « r ») et le temps de calcul moyen en secondes à la fin des étapes correspondantes (l'axe « $temps, s$ »).

FIGURE 3.3 – Dynamique du processus de résolution pour la série 75_1.



Pour la série 75_1, l'heuristique gloutonne donne des solutions avec une erreur relative moyenne de 26% en 3 minutes de calcul (ce temps inclue aussi la formation des LA-stations), et 1000 applications de l'heuristique gloutonne randomisée fournit une erreur relative moyenne

TABLEAU 3.6 – Répartition des instances pour la famille 1

Séries	Nombre de LA-stations		
	$< 0,5 \cdot 10^6$	$[0,5 \cdot 10^6; 1,1 \cdot 10^6]$	$> 1,1 \cdot 10^6$
75_1	15	5	0
75_2	20	0	0
75_3	20	0	0
75_4	20	0	0
95_1	7	12	1
95_2	20	0	0
95_3	20	0	0
95_4	20	0	0
115_1	3	10	7
115_2	20	0	0
115_3	20	0	0
115_4	20	0	0
Total	205	27	8

de 11% pour le temps de calcul de 4 minutes. La performance des heuristiques pour les autres séries est similaire.

Dans les Tableaux 3.6 et 3.7 chaque élément montre combien d'instances d'une série (définie par une ligne) ont le nombre de LA-stations dans un intervalle donné (défini par la colonne correspondante). Rappelons que le nombre de LA-stations correspond au nombre de variables binaires dans les problèmes MinSPP et Relax-MinSPP.

Les Tableaux 3.8 et 3.9 montrent le nombre d'instances résolues à l'optimalité et approximativement (les lignes « Opt. » et « Appr. », respectivement). Pour les instances résolues approximativement, les erreurs relatives moyennes en pourcentage sont données entre parenthèses.

Les expérimentations ont montré que normalement les instances avec le nombre de LA-stations inférieur à 0,5 millions sont facilement résolues à l'optimalité. Lorsque le nombre de LA-stations est inférieur à 1,1 millions, il y a une bonne chance pour trouver une solution

TABLEAU 3.7 – Répartition des instances pour la famille 2

Séries	Nombre de LA-stations		
	$< 10^5$	$[10^5; 5 \cdot 10^5]$	$> 5 \cdot 10^5$
70_1	19	1	0
70_2	20	0	0
70_3	20	0	0
90_1	10	10	0
90_2	20	0	0
90_3	20	0	0
110_1	3	17	0
110_2	20	0	0
110_3	20	0	0
130_1	0	12	8
130_2	19	1	0
130_3	20	0	0
Total	191	41	8

TABLEAU 3.8 – Qualité de solutions pour la famille 1

Statut de	Nombre de LA-stations		
solution	$< 0,5 \cdot 10^6$	$[0,5 \cdot 10^6; 1,1 \cdot 10^6]$	$> 1,1 \cdot 10^6$
Opt.	191	7	0
Appr. (moy. r)	14 (5,6%)	20 (7,1%)	6 (N/A)

TABLEAU 3.9 – Qualité de solutions pour la famille 2

Statut de solution	Nombre de LA-stations		
	$< 10^5$	$[10^5; 5 \cdot 10^5]$	$> 5 \cdot 10^5$
Opt.	190	28	3
Appr. (moy. r)	1 (0,5%)	13 (2,2%)	5 (4,1%)

approchée de haute qualité et donner une erreur relative. Pour les instances avec plus de 1,1 millions LA-stations, des solutions approchées peuvent être trouvées mais des bornes inférieures ne peuvent pas être obtenues à cause du débordement de mémoire utilisé par le solveur.

Notons que les instances de la famille 2 ont moins de LA-stations en raison des ensembles de blocs plus petits. Donc, il est plus facile de les résoudre. Dans la plupart des cas, les solutions optimales ont été trouvées. Dans les autres cas, les solutions admissibles obtenues sont de bonne qualité.

Pour les séries de la famille 1 et 2, les Tableaux 3.10 et 3.11 montrent : le nombre d'instances résolues dans l'intervalle de temps (en heures) défini par les colonnes ; les temps de résolution moyens en secondes (colonne « temps ») ; le nombre d'itérations de l'algorithme de génération de contraintes (a.g.c.) appliqué à l'ensemble original $\mathcal{W}$ (colonne « # Itér. ») ; le nombre d'instances résolues à l'optimalité (colonne « # Opt. ») ; le nombre de solutions approchées (colonne « # Fais. »).

Les résultats numériques pour la famille 2 sont donnés dans le Tableau 3.11. Rappelons que ces instances ont été utilisées dans [Delorme et al., 2012] pour tester l'approche de la clique de poids maximal. Les séries 70_1, 70_2, et 70_3 ont été les plus grandes considérées dans [Delorme et al., 2012]. En employant l'approche de clique de poids maximal, les solutions optimales ont été trouvées pour la plupart des instances sauf pour les séries 60_1, 70_1, et 70_2. Dans les séries 60_1, les solutions optimales n'ont été trouvées que pour 8% des instances, dans les séries 70_2 ce rapport a été 16%. Aucune instance de la série 70_1 n'a été résolue à l'optimalité. En outre, pour 8% des instances dans 60_1 et 18% des instances dans 70_1, même aucune solution admissible n'a été obtenue. Bien que ces expérimentations aient été effectuées

TABLEAU 3.10 – Résolution des séries de la famille 1 par a.g.c.

Séries	Temps de résolution				temps, s.	# Itér.	# Opt.	# Appr.
	<0,5h	[0,5h ; 1h]	]1h ; 1,5h]	>1,5h	(moy.)	(moy.)		
75_1	9	5	2	4	2443	4,6	13	7
75_2	18	1	0	1	587	4,3	18	2
75_3	20	0	0	0	54,3	8,6	20	0
75_4	20	0	0	0	0,78	3,0	20	0
95_1	3	8	8	1	3320	2,9	6	14
95_2	17	2	0	1	1237	5,4	20	0
95_3	20	0	0	0	32,5	4,5	20	0
95_4	20	0	0	0	1,55	3,5	20	0
115_1	0	6	5	7	5431	2,5	4	14
115_2	14	2	1	3	2040	8,4	17	3
115_3	20	0	0	0	97,4	7,1	20	0
115_4	20	0	0	0	2,6	3,2	20	0

TABLEAU 3.11 – Résolution des séries de la famille 2 par a.g.c.

Séries	Temps de résolution				temps, s.	# Itér.	# Opt.	# Appr.
	<0,5h	[0,5h ; 1h]	]1h ; 1,5h]	>1,5h	(moy.)	(moy.)		
70_1	20	0	0	0	201	4,1	19	1
70_2	20	0	0	0	3,49	3,0	20	0
70_3	20	0	0	0	0,33	1,7	20	0
90_1	19	0	0	1	861	3,2	16	4
90_2	20	0	0	0	37,5	4,0	20	0
90_3	20	0	0	0	0,96	2,2	20	0
110_1	13	1	6	0	1982	4,7	14	6
110_2	18	2	0	0	185	5,3	19	1
110_3	20	0	0	0	2,92	3,7	20	0
130_1	3	13	3	1	3113	3,3	13	7
130_2	19	1	0	0	60,13	4,6	20	0
130_3	20	0	0	0	3,46	3,4	20	0

sur un ordinateur avec une vitesse de processeur plus faible par rapport à celui qui a été utilisé dans notre cas, la limite de temps (3 heures) et le volume de mémoire à accès direct (16 Go) ont été plus grands.

Nos expérimentations, présentées dans le Tableau 3.11, montrent la capacité de l'algorithme de génération de contraintes à trouver des solutions optimales même pour des problèmes plus grands en temps plus court. Pour les instances de la famille 2, l'algorithme de génération de contraintes n'a jamais échoué à trouver au moins une solution admissible.

Le Tableau 3.12 contient les résultats de l'application directe de Cplex pour le problème P dans la formulation de programme en nombres entiers mixtes proposée dans [Belmokhtar et al., 2006]. Pour les séries de la famille 1, que nous avons utilisé comme une extension des données de référence de [Belmokhtar et al., 2006], le Tableau 3.12 montre : le nombre d'instances résolues dans les intervalles de temps (en heures) ; les temps de résolution moyens ; le nombre de solutions optimales ; le nombre de solutions approchées. La dernière co-

lonne (« # Échecs ») montre le nombre d'instances dans chaque série pour lesquelles aucune solution admissible n'a été trouvée à cause de la limite de mémoire ou de temps.

Afin de faire une comparison, le temps de calcul a été limité à 2 heures comme pour notre algorithme de génération de contraintes. Notons que ces résultats sont meilleurs par rapport à ceux qui ont été présentés dans [Belmokhtar et al., 2006] grâce à l'emploi d'une version de Cplex plus récente. On peut voir que pour les instances de petite taille les performances de l'algorithme de génération de contraintes et de Cplex sont comparables. Pour les instances de grande taille (les séries 95_1, 115_1, 115_2, 115_3) Cplex, dans de nombreux cas, a échoué à trouver même une solution admissible, tandis que l'algorithme de génération de contraintes a trouvé des solutions optimales dans beaucoup de cas et n'a échoué à trouver une solution admissible que pour deux instances de la série 115_1. La comparaison des temps de calcul montre aussi l'avantage de l'algorithme de génération de contraintes.

Tableau 3.12 – Application directe de Cplex pour les séries de la famille 1

Séries	Temps de résolution				Moy.	# Opt.	# Appr.	# Échecs
	<0,5h	[0,5h ; 1h]	[1h ; 1,5h]	>1,5h	temps, s.			
75_1	13	3	2	2	1769	18	2	0
75_2	15	1	0	4	1920	16	4	0
75_3	13	5	0	2	1582	18	2	0
75_4	20	0	0	0	46	20	0	0
95_1	0	1	4	9	4602	9	5	6
95_2	7	2	3	8	3998	7	13	0
95_3	13	4	1	2	1919	19	1	0
95_4	20	0	0	0	117	20	0	0
115_1	0	0	0	1	7200	0	1	19
115_2	0	0	0	0	–	0	0	20
115_3	1	0	0	1	3820	0	2	18
115_4	20	0	0	0	135	20	0	0

On peut observer que 81% de toutes les instances considérées ont été résolues en 30 minutes et 91% ont été résolues en 1 heure. L'approche de résolution que nous proposons surpasse les approches antérieures de [Belmokhtar et al., 2006] et [Delorme et al., 2012]. La taille de problème en termes de données d'entrée et le pourcentage de solutions optimales ont considérablement augmenté par rapport aux études précédentes.

Nos expérimentations ont montré que les instances avec la densité de contraintes de précédence faible sont les plus difficiles à résoudre en raison du grand nombre de LA-stations. Dans ce cas, l'algorithme de génération de contraintes peut s'arrêter après un petit nombre d'itérations parce que la limite de mémoire ou de temps est dépassée. Notons également que dans les lancements de Cplex qui ont mené à une solution admissible du Relax-MinSPP, il n'y a pas eu de nombreux cycles et le nombre d'itérations a été faible.

Pour les instances avec une densité de contraintes de précédence moyenne nous avons eu de 100000 jusqu'à 300000 LA-stations. Logiquement, pour trouver une solution acyclique pour ces instances, il faut faire plus d'itérations. Pourtant, ces itérations sont rapides et la plupart des instances ont été résolues à l'optimalité.

Pour les instances avec une densité de contraintes de précédence élevée nous avons eu au plus 100000 LA-stations et elles ont été résolues en 10 minutes environ. Les instances avec la densité très élevée de la famille 1 ont moins de 2000 LA-stations et elles sont résolues à l'optimalité en quelques secondes.

3.10 Conclusion

Dans ce chapitre nous avons présenté un problème de configuration de lignes de fabrication dédiées. Ce problème comporte à la fois des décisions concernant le choix de ressources et leur affectation aux stations. La ligne est cadencée et son schéma logique est linéaire : les stations sont installées l'une après l'autre. Il y a un seul cheminement de production. Les pièces de produit sont transportées le long de la ligne dans la même direction et elles subissent des opérations jusqu'à leur transformation en produits finis.

Les ressources sont regroupées en ensemble de blocs. Un bloc exécute simultanément un ensemble d'opérations. Dans l'industrie de tels blocs sont, par exemple, des boîtiers multibroches

qui effectuent des opérations d'usinage. Nous examinons un environnement de production où les blocs de la même station sont activés en parallèle, ce qui augmente davantage la productivité de la ligne. Nous supposons que si une borne supérieure sur le temps de cycle est donnée, alors le temps opératoire le plus long, qui détermine le temps de cycle, ne dépasse pas cette borne.

Le problème comporte des contraintes d'affectation des ressources, à savoir : les contraintes de précédence, d'exclusion et d'inclusion. En outre, il y a des bornes supérieures sur le nombre de stations et le nombre de blocs par station.

Le modèle de ce problème, dénommé le problème P, est orienté coût. Chaque bloc a un coût et l'installation d'une station suscite un coût complémentaire constant. Ainsi, le problème P implique la sélection d'un sous-ensemble des blocs disponibles et l'affectation de ces blocs aux stations de sorte que le coût total soit minimal.

Afin de résoudre le problème P, nous avons proposé une approche qui se base sur deux éléments : le concept d'une station localement admissible et la réduction du problème P à un problème de partition d'ensemble. Plusieurs idées de prétraitement des données visant à la réduction du nombre de variables et de la taille de l'espace des solutions admissibles ont été présentées. Deux heuristiques gloutonnes et un algorithme de génération de contraintes ont été développés pour résoudre le problème P. Les expérimentations numériques avec les données de référence ont montré que l'approche proposée surpasse les méthodes de résolution antérieures à la fois en termes de qualité des solutions et d'effort de calcul.

Chapitre 4

Minimisation des coûts de changements de séries pour des lignes multi-produits : cas d'exécution parallèle des opérations

4.1 Introduction

Dans ce chapitre et le chapitre 5 nous allons aborder deux problèmes qui concernent la configuration d'une ligne multi-produit avec des coûts de changements de séries. La différence entre ces deux problèmes réside dans le mode d'exécution des opérations sur les stations. D'abord, nous allons examiner un problème avec le mode d'exécution des opérations parallèle comme dans le chapitre 3. On peut notamment trouver des exemples d'exécution parallèle des opérations dans l'industrie mécanique [Belmokhtar et al., 2006, Dolgui et al., 2008] et dans l'industrie automobile [Falkenauer, 2005].

4.2 Définition du problème

Les hypothèses du problème étudié seront décrites en utilisant la classification à six éléments du chapitre 2.

La **variabilité de ligne**. Une ligne doit être configurée pour la production de masse de f types de produits. La ligne est donc *flexible* ou *reconfigurable*.

Soit $F = \{1, \ldots, f\}$ l'ensemble des types de produits. Chaque produit de type $v \in F$ nécessite

l'exécution une seule fois sur la ligne d'un ensemble donné N_v d'opérations. Les opérations de l'ensemble N_v sont appelées les *opérations de type* v. Différents ensembles N_v peuvent contenir des opérations communes. Ainsi, la même opération peut être de différents types. Soit $N := \cup_{v=1}^{f} N_v = \{1, \ldots, n\}$ le surensemble de toutes les opérations nécessaires, et soit T_i l'ensemble des types de l'opération $i \in N$, c'est-à-dire, $T_i = \{v \mid i \in N_v,\ v \in F\}$. La réaffectation d'opérations (rééquilibrage) n'est pas permise, c'est-à-dire que le positionnement des ressources dans la ligne reste le même quelque soit le type de produit traité.

Les **caractéristiques de ligne**. La ligne est cadencée et le *cheminement de production est unique* car il n'y a pas de stations parallèles. Le *schéma logique de la ligne est linéaire* : les produits de tous les types sont transportés entre des stations dans la même direction.

Les **caractéristiques de temps**. Les *temps opératoires sont déterministes*. Vu qu'il y a plusieurs types de produits, les temps de cycle différent selon le type de produit. Nous supposons qu'il y a une *borne supérieure commune sur les temps de cycle* des produits de tous les types. Cette borne est donc appliquée pour toutes les stations. En raison du mode d'exécution parallèle des opérations, le temps de cycle est égal au temps opératoire le plus grand plus le temps de changement de séries correspondant parmi toutes les opérations de tous les types. Nous supposons que cette borne supérieure commune n'est excédée par aucun temps de cycle. Ainsi, les temps opératoires et les temps de changements de séries ne sont pas présents dans la formulation du problème et les temps de cycle ne sont pas considérés.

Les **caractéristiques de coût**. S'il y a au moins une opération d'un certain type affectée à une station, alors le changement de séries aura lieu sur cette station. Un coût de changement de séries a_v est associé au type de produits v, $v = 1, \ldots, f$. Les coûts de changements de séries sont liés avec les démarches nécessaires pour commencer et finir le traitement d'un produit sur une station. Comme nous l'avons mentionné dans le chapitre 1, ces démarches incluent : le chargement du produit, le positionnement du produit et des ressources/outils, le déchargement et le nettoyage de la station.

Les **contraintes technologiques**. Le *mode d'exécution des opérations* sur les stations est **parallèle**.

Les *contraintes de précédence* sont données sur le surensemble N. Si une opération $i \in N$

précède une opération $j \in N$, alors l'opération j ne peut pas être affectée à la station de i ou à n'importe quelle station qui la précède. Les relations de précédence sont transitives et irréflexives. Elles sont représentées par un graphe orienté acyclique $G = G(N, A)$, dans lequel il y a un arc $(i, j) \in A$ si et seulement si une opération i précède une opération j. Les relations de précédence pourraient être représentées séparément pour chaque type d'opérations mais vu que le rééquilibrage n'est pas permis, elles doivent pouvoir être représentées par un graphe acyclique sur le surensemble N.

Les *contraintes d'exclusion* sont définies sur le surensemble N. Elles peuvent être représentées par une collection E de sous-ensembles $E' \subset N$ telle que toutes les opérations de E' ne peuvent pas être affectées à la même station, mais n'importe quel sous-ensemble propre de E' peut être affecté à la même station. Ces relations interdisent l'utilisation simultanée des outils qui sont en conflit à cause de leurs caractéristiques physiques. Supposons, sans perte de généralité, qu'il n'y a pas de sous-ensembles dans E qui se contiennent les uns les autres, et qu'il n'y a pas deux opérations du même ensemble d'exclusion qui se précédent l'une l'autre.

Chaque opération $i \in N$ a sa *taille* s_i qui représente le nombre d'outils standard nécessaires pour effectuer cette opération. La taille totale de toutes les opérations affectées à la même station (le nombre d'outils standard) ne doit pas dépasser la *capacité de la station* r. Vu que chaque outil standard est associé à une opération élémentaire, chaque opération i peut être considérée comme un bloc des opérations élémentaires qui doivent être simultanément exécutées sur la même station. Ainsi, s_i représente la cardinalité de ce bloc et N représente l'ensemble des blocs. Donc, des opérations de différentes tailles sont similaires aux *ensembles d'inclusion* du chapitre 3. Les ensembles d'inclusion sont utilisées pour des opérations élémentaires dont la précision peut être perdue si elles sont affectées à des stations différentes.

Les **critères d'optimisation**. Soit x_v le nombre de changements de séries pour un produit de type v, $v = 1, \ldots, f$. Une décision doit être prise à propos du nombre de stations, k, et de l'affectation des opérations aux stations $1, \ldots, k$.

Le problème a deux critères. Le critère principal est la *minimisation du nombre de stations*, k. Le critère secondaire est la *minimisation du coût total de changements de séries*, $a_1 x_1 + \cdots + a_f x_f$.

Nous désignons le problème avec seulement le premier critère comme $P_{par}(prec, excl, s_i \mid k)$ et le problème avec deux critères ordonnés comme $P_{par}(prec, excl, s_i \mid k, cost)$. Notons que ces deux critères ne sont pas redondants, c'est-à-dire, la minimisation du coût total de changements de séries n'est pas suffisante pour la minimisation du nombre de stations et inversement. Soit (k^*, c^*) le minimum lexicographique des valeurs des fonctions objectifs pour le problème $P_{par}(prec, excl, s_i \mid k, cost)$. Notons que l'optimisation lexicographique consiste à minimiser le premier objectif k et ensuite, pour le nombre des stations optimal k^*, à minimiser le deuxième objectif, c'est-à-dire, trouver la valeur minimale c^* du coût total de changements de séries.

4.3 Origine du problème

Nous avons observé un tel problème dans les entreprises PCI-SCEMM (Saint-Étienne, France) et MZAL (Minsk, Biélorussie) qui fabriquent des lignes d'usinage automatiques. Leurs lignes sont destinées pour la production de masse de plusieurs variantes (types) d'une pièce. Les coûts de changements de série sont estimés en analysant l'ensemble de dépenses liées aux démarches nécessaires pour préparer une station au traitement d'une pièce d'un certain type. Chaque station est équipée d'un boîtier multibroche où des outils pour perçage, meulage, fraisage, etc. sont montés pour exécuter plusieurs opérations en parallèle. Les résultats de ces opérations sont des trous plats, filetés ou chanfreinés de différentes profondeurs et diamètres.

Un magasin contenant les outils nécessaires est associé à chaque boîtier multibroche. Tous les magasins et les boîtiers multibroches ont la même capacité maximale, r. Lorsqu'un produit d'un type spécifique arrive à une station, tous les outils inutiles sont enlevés du boîtier multibroche et sont retournés au magasin. Ensuite, les outils nécessaires sont pris du magasin et sont montés sur le boîtier. Il y a des relations de précédence et d'exclusion entre les opérations. Le traitement d'un produit sur une station implique la présence d'un opérateur ainsi qu'une consommation d'énergie qui suscitent des coûts élevés. Donc, le problème consiste à affecter des outils (opérations) aux stations de sorte que le nombre des stations soit minimal. De plus, pour le nombre minimal de stations, le coût total des changements de séries doit aussi être minimisé.

4.4 Exemple d'une solution

Une solution pour ce problème peut être représentée comme un tableau dont les colonnes représentent des stations et un rectangle dans la colonne représente une opération i suivie par ces types (ensemble T_i) entre parenthèses. La taille d'une opération est le nombre de lignes dans un rectangle. Le Tableau 4.1 montre une solution du problème avec 4 stations de capacité $r = 6$ et 15 opérations de trois types. Le coût total de cette solution est $3a_1 + 3a_2 + 2a_3$ parce que les types 1 et 2 sont présents sur trois stations tandis que le type 3 est sur deux stations.

Tableau 4.1 – Exemple d'une solution

<table>
<tr><th></th><th>Station 1</th><th>Station 2</th><th>Station 3</th><th>Station 4</th></tr>
<tr><td>1</td><td rowspan="3">2(1,2,3)</td><td>1(1,2)</td><td rowspan="2">3(1,3)</td><td>12(2)</td></tr>
<tr><td>2</td><td rowspan="3">4(1,2)</td><td>13(2)</td></tr>
<tr><td>3</td><td>5(3)</td><td>14(2)</td></tr>
<tr><td>4</td><td rowspan="2">7(1,2,3)</td><td>6(3)</td><td>15(2)</td></tr>
<tr><td>5</td><td rowspan="2">10(1,2)</td><td>8(1)</td><td rowspan="2">-</td></tr>
<tr><td>6</td><td>9(2,3)</td><td>11(1)</td></tr>
</table>

4.5 Minimisation du nombre de stations

Vu que le problème étudié a deux critères, nous allons successivement considérer chaque critère. Le problème à un seul critère, désignons le comme $P_{par}(prec, excl, s_i \mid k)$, consiste à trouver le nombre minimal de stations. Le cas le plus simple avec des opérations de taille unitaire sans contraintes d'exclusion et de précédence peut être facilement résolu : $k^* = \lceil \frac{n}{r} \rceil$. Dans les sous-sections suivantes, les cas plus complexes sont considérés.

4.5.1 Cas avec des contraintes de précédence et des tailles unitaires sans contraintes d'exclusion

Le cas particulier de cette sous-section est noté comme $P_{par}(prec, s_i = 1 \mid k)$. Nous allons montrer qu'il est équivalent au problème d'ordonnancement $Pm \mid p_i = 1, prec \mid C_{\max}$. La

description de ce problème est la suivante.

Il y a n tâches de temps unitaire qui doivent être ordonnées sur m machines identiques parallèles. Des contraintes de précédence sont données sur l'ensemble des tâches de sorte que si une tâche i précède une tâche j, alors j ne peut être exécutée ni dans l'intervalle de temps unitaire de i ni avant cette intervalle. Aucune préemption du traitement de tâche n'est permise et aucune paire de tâches ne peut pas être simultanément exécutée sur la même machine. L'objectif est d'affecter des tâches aux intervalles de temps unitaires sur les machines de sorte que les contraintes de précédence sont satisfaites et que le temps d'achèvement de la dernière tâche, $C_{\max}$, soit minimisé. Désignons la valeur minimale de $C_{\max}$ comme $C^*_{\max}$.

Pour le problème $P_{par}(prec, s_i = 1 \mid k)$, nous interprétons les opérations comme des tâches, la borne de capacité r comme le nombre de machines m, et les stations $1, 2, \ldots$ comme des intervalles de temps unitaire $[0, 1], [1, 2], \ldots$ dans le problème $Pm \mid p_i = 1, prec \mid C_{\max}$. On peut voir que toute solution admissible du problème $P_{par}(prec, s_i = 1 \mid k)$ avec un nombre de stations k peut être représentée comme une solution admissible du problème d'ordonnancement correspondant avec la valeur objective $C_{\max} = k$ et vice versa. Donc, ces deux problèmes sont équivalents et $k^* = C^*_{\max}$.

Si les contraintes de précédence sont arbitraires, alors le problème $Pm \mid p_i = 1, prec \mid C_{\max}$ est NP-difficile au sens fort pour un nombre de machines variable m, voir [Ullman, 1975], et son statut de complexité est ouvert si m est une constante supérieure à deux, voir [Brucker and Knust, 2006]. Le problème $Pm \mid p_i = 1, prec \mid C_{\max}$ peut être résolu en temps $O(n)$ si le graphe de précédence, G, est une collection d'arbres entrant (in-trees en anglais), voir [Hu, 1961], ou une collection d'arbres sortant (out-trees), voir [Davida and Linton, 1976]. Il peut être résolu en temps $O(n^2)$ si G est arbitraire et $m = 2$ (l'algorithme le plus connu est décrit dans [Gabow, 1982]).

Ainsi, le problème $P_{par}(prec, s_i = 1 \mid k)$ est NP-difficile au sens fort pour une borne de capacité r quelconque et un graphe de précédence arbitraire G. Il peut être résolu en temps polynomial si G est une collection d'arbres entrant ou une collection d'arbres sortant, ou si $r = 2$.

Des relations de précédence en forme de chaîne et en forme d'arbre sont typiques pour

des opérations d'assemblage et d'usinage. Nous allons donc donner notre interprétation de l'algorithme de Hu [Hu, 1961] pour le problème avec des relations de précédence en arbres entrant. Ici, les termes « opération » et « sommet » sont utilisés de façon interchangeable. Nous remplaçons la notation *prec* par $in-tree$ ou $out-tree$ pour les problèmes avec des relations de précédence en arbres entrant ou en arbres sortant, respectivement. Numérotons les stations $t = 1, 2, \ldots$

Algorithme $NPHD$ (niveau le plus haut d'abord) pour le problème $P_{par}(in-tree, s_i = 1 \mid k)$

1. **Préparation des données d'entrée**. Pour chaque sommet $j \in N$, stocker ses prédécesseurs directs, leur nombre, et son unique successeur direct. Calculer le *niveau* (le nombre total de successeurs indirects plus le successeur direct) l_j de chaque sommet $j \in N$ en scannant les sommets des racines aux feuilles : les racines obtiennent le niveau zéro, leurs prédécesseurs directs obtiennent le niveau un, etc. Avec chaque sommet $j \in N$, stocker son niveau l_j. Soit $l_{\max} = \max\{l_j | j \in N\}$. Pour chaque $l = 0, 1, \ldots, l_{\max}$, former la liste L_l des sommets qui ont le niveau l et n'ont pas de prédécesseurs. Former la liste des sommets $L = (L_{l_{max}}, L_{l_{\max}-1}, \ldots, L_0)$ qui n'ont pas de successeurs et qui apparaissent dans l'ordre non-croissant de leurs niveaux. Certaines listes L_l dans L peuvent être vides. Mettre $t = 1$.
2. **Affectation des opérations**. Affecter la première opération (avec le plus haut niveau) de la liste L à une station t. Si toutes les opérations de l'ensemble N sont affectées, alors arrêter - une solution optimale est construite et $k^* = t$. Sinon, mettre à jour la liste L en enlevant sa première opération. Diminuer d'une unité le nombre de prédécesseurs du successeur direct de l'opération affectée. Si le nombre de prédécesseurs d'un sommet de niveau l est devenu égal à zéro, alors stocker ce sommet dans une liste auxiliaire Z_l. Si toutes les opérations de la liste L sont affectées ou si toutes les r positions de la station t sont occupées, alors mettre à jour la liste L en mettant $L_l := (L_l, Z_l)$ pour chaque liste auxiliaire Z_l. Remettre $t := t + 1$ et répéter la première étape de l'algorithme.

L'algorithme $NPHD$ fonctionne en temps $O(n)$. Le même algorithme peut être utilisé pour résoudre le problème $P_{par}(out-tree, s_i = 1 \mid k)$ si on inverse les contraintes de précédence et

on lit la solution optimale obtenue pour le problème $P_{par}(in-tree, s_i = 1 \mid k)$ de la dernière station à la première.

4.5.2 Cas avec des tailles d'opérations arbitraires sans contraintes d'exclusion et de précédence

Notons le cas particulier traité dans cette sous-section comme $P_{par}(s_i \mid k)$. Le problème $P_{par}(s_i \mid k)$ peut être interprété comme un problème de bin packing, dans lequel il y a n articles de tailles $s_1, \ldots, s_n$ qui doivent être placés dans le nombre minimal de boîtes ayant la même capacité r. Le nombre minimal de stations k^* est égal au nombre minimal de boîtes nécessaires.

Le problème de bin packing est NP-difficile au sens fort. Il peut être résolu en temps $O(n^{K^L})$ s'il y a au plus K tailles d'articles distinctes et au plus L articles peuvent être placés dans une boîte, voir, par exemple, [Fernandez de la Vega and Lueker, 1981]. Si r est une constante, alors K et L sont aussi des constantes, et le problème de bin packing peut être résolu en temps polynomial. Il peut être réduit à une résolution de n problèmes d'ordonnancement de machines parallèles $Pm \mid d_j = r \mid \cdot$, $m = 1, \ldots, n$. Dans le problème $Pm \mid d_j = r \mid \cdot$, il y a n tâches non-préemptives avec les temps de traitement $s_1, \ldots, s_n$. Ces tâches doivent être ordonnées sur m machines parallèles identiques. La question est de savoir si toutes les tâches peuvent être accomplies avant la date limite r. Nous avons $k^* = m^*$, où m^* est la valeur minimale de m pour laquelle la question du problème $Pm \mid d_j = r \mid \cdot$ a une réponse affirmative. Le problème $Pm \mid d_j = r \mid \cdot$ peut être résolu en temps $O(nmr^{m-1})$ par un algorithme de programmation dynamique. Par conséquent, le problème $P_{par}(s_i \mid k)$ peut être résolu en temps $O(n^2 r^{n-1} \log_2 n)$ par une méthode de dichotomie en utilisant $1, \ldots, n$ des valeurs possibles de m.

Ainsi, nous en déduisons que le problème $P(s_i \mid k)$ est NP-difficile au sens fort. Il peut être résolu en temps polynomial si le nombre d'opérations est une constante ou si la capacité des stations r est une constante.

4.5.3 Cas avec des contraintes d'exclusion et sans contraintes de précédence

Notons le cas particulier étudié dans cette sous-section comme $P_{par}(excl, s_i = 1 \mid k)$.

Théorème 1 *Le problème* $P_{par}(excl, s_i = 1 \mid k)$ *est NP-difficile au sens fort si* $|E'| = 2$ *pour chaque* $E' \in E$.

Preuve: Nous utilisons une réduction du *problème de la couverture exacte par 3-ensembles (X3C)*, voir [Garey and Johnson, 1979]. Le dernier problème est le suivant. En ayant une famille $A = \{A_1, \ldots, A_a\}$ de sous-ensembles à 3 éléments de l'ensemble $B = \{1, \ldots, 3b\}$, est-ce que A contient une couverture exacte de B, c'est-à-dire, une sous-famille $S \subseteq A$ telle que chaque $i \in B$ appartient à exactement un ensemble à 3 éléments dans S ? Supposons que $a > b$, car sinon, une solution triviale existe.

En ayant une instance de *X3C*, construisons l'instance suivante de la version de décision du problème $P_{par}(excl, s_i = 1 \mid k)$. Définissons le nombre d'opérations $n = (b+1)(a-b+1)$ et la capacité des stations $r = b + 1$. L'ensemble des opérations inclut :

1. Une seule *X-opération*.
2. Le nombre b des mêmes *Y_i-opérations*, $i = 1, \ldots, a - b$.
3. Des *A_i-opérations* associées aux ensembles A_i, $i = 1, \ldots, a$.

Supposons que toutes les opérations sont de même type. Les contraintes d'exclusion sont représentées de la façon suivante. La X-opération exclut toute Y_i-opération, c'est-à-dire que $\{X, j\} \in E$ pour chaque Y_i-opération j, $i = 1, \ldots, a - b$. Toute Y_i-opération exclut toute Y_q-opération, c'est-à-dire, $\{j, l\} \in E$ pour chaque Y_i-opération j et chaque Y_q-opération l, $i \neq q$. L'A_i-opération et l'A_q-opération s'excluent mutuellement si les sous-ensembles A_i et A_q partagent le même élément, c'est-à-dire, si $A_i \cap A_q \neq \phi$, $i \neq q$.

Nous allons montrer que pour l'instance construite, une solution de *X3C* existe *si et seulement si* il existe une solution admissible pour cette instance du problème $P(excl, s_i = 1 \mid k)$ avec au plus $a - b + 1$ stations. Il est évident que la construction d'instance est pseudo-polynomial.

Partie « si ». Supposons qu'il existe une solution admissible pour l'instance du problème $P_{par}(excl, s_i = 1 \mid k)$ avec au plus $a-b+1$ stations. Nous observons qu'il y a $a-b+1$ opérations mutuellement exclusives : la X-opération et une Y_i-opération pour chaque $i = 1, \ldots, a - b$. Par conséquent, il doit y avoir exactement $a-b+1$ stations. Sans perte de généralité, nous supposons qu'une Y_i-opération est affectée à la station i, $i = 1, \ldots, a-b$, et que la X-opération est affectée

à la station $a-b+1$. Vu que toute Y_i-opération exclut toute Y_q-opération pour $i \neq q$, et qu'il y a $a-b+1$ stations, toutes les Y_i-opérations doivent être à la station i, $i = 1, \ldots, a-b$. Donc, l'A_i-opération, $i \in \{1, \ldots, a\}$, peut être affectée à une des b positions restantes de la station $a-b+1$ ou à la seule position de n'importe quelle station sauf à la station $a-b+1$. La structure d'une telle solution est donnée dans le Tableau 4.2, où les colonnes représentent les stations et les symboles « • » représentent les A_i-opérations.

TABLEAU 4.2 – Structure d'une solution admissible avec $a-b+1$ stations.

Y_1	Y_2	•	$\cdots$	Y_{a-b}	X
•	Y_2	Y_3	$\cdots$	Y_{a-b}	•
Y_1	•	Y_3	$\cdots$	•	•
Y_1	Y_2	Y_3	$\cdots$	Y_{a-b}	•
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
Y_1	Y_2	Y_3	$\cdots$	Y_{a-b}	•

Les b A_i-opérations sur la station $a-b+1$ doivent être mutuellement non-exclusives, c'est-à-dire, aucune paire des sous-ensembles A_i et A_q correspondant à ces opérations ne doit partager le même élément. Nous déduisons que les sous-ensembles A_i correspondant aux A_i-opérations affectées à la station $a-b+1$ constituent une couverture exacte de B. Cela accomplit la preuve de la partie « si ».

Partie « seulement si ». S'il existe une solution S du problème *X3C*, alors construisons une solution du problème $P(excl, s_i = 1 \mid k)$ avec la structure donnée dans le Tableau 4.2 et affectons l'A_i-opération à la station $a-b+1$ si et seulement si $A_i \in S$. Une telle solution est admissible et le nombre de stations est égal à $a-b+1$. Cela accomplit la preuve de la partie « seulement si ». ∎

4.5.4 Heuristiques pour le problème $P_{par}(prec, excl, s_i \mid k)$

Dans les sous-sections précédentes nous avons montré que le problème à un seul critère $P_{par}(prec, excl, s_i \mid k)$ est NP-difficile au sens fort si une des trois conditions suivantes est satisfaite :

1. Les tailles des opérations ne sont pas unitaires.
2. Il y a des contraintes de précédence.
3. Il y a des contraintes d'exclusion.

Maintenant nous allons aborder le problème général $P_{par}(prec, excl, s_i \mid k)$. Afin de le résoudre, nous proposons une heuristique gloutonne et sa version randomisée. Numérotons les stations comme suit $t = 1, 2, \ldots$

Heuristique $GreedyNumberPar1$ **pour le problème** $P_{par}(prec, excl, s_i \mid k)$

1. **Élimination des contraintes d'exclusion.** S'il n'y a aucun ensemble d'exclusion ($E = \phi$), alors mettre $t = 1$ et passer à l'étape 2. Sinon, sélectionner un ensemble $E' \in E$. De façon aléatoire choisir une paire d'opérations dans E' et introduire une relation de précédence entre elles. Notons que si cette relation de précédence est satisfaite, alors la relation d'exclusion donnée par E' est aussi vérifiée. Mettre à jour E en retirant E' et toute autre ensemble d'exclusion qui contient une paire des opérations avec un chemin orienté entre elles dans le nouveau graphe G. Répéter l'étape 1.
2. **Affectation des opérations en résolvant le problème de la somme de sous-ensembles.** Calculer l'ensemble N^+ des sommets du graphe de précédence G qui n'ont pas de prédécesseurs. Affecter à la station t les opérations d'un sous-ensemble $X \in N^+$ qui est sélectionné par la résolution du problème de la somme de sous-ensembles suivant :

$$\max_{X \in N^+} \sum_{i \in X} s_i, \text{ sous la contrainte } \sum_{i \in X} s_i \leq r.$$

   Mettre à jour le graphe G en retirant les sommets de l'ensemble X et leurs arcs sortants. Si G est vide, alors arrêter : une solution admissible est construite. Sinon, remettre $t := t+1$ et répéter l'étape 2.

La résolution itérative du problème de la somme de sous-ensembles a été prouvé être une approche efficace pour les problèmes de bin packing, voir, par exemple, [Caprara and Pferschy, 2005] et [Haouari and Serairi, 2009]. Afin de résoudre le problème de la somme de sous-ensembles, nous utilisons l'algorithme de programmation dynamique standard qui nécessite un temps $O(|N^+|r)$.

L'heuristique $GreedyNumberPar1$ peut être appliquée plusieurs fois pour augmenter la probabilité d'obtenir une solution admissible avec un nombre de station faible.

L'heuristique $GreedyNumberPar2$ se distingue de $GreedyNumberPar1$ en ce qu'à l'étape 2, au lieu de résoudre un problème de la somme de sous-ensembles, le sous-ensemble $X \in N^+$ tel que $\sum_{i \in X} s_i \leq r$ est généré par la sélection de ses éléments de manière randomisée.

4.5.5 Formulation en un programme linéaire en nombres entiers du problème $P_{par}(prec, excl, s_i \mid k)$

Afin de résoudre le problème $P_{par}(prec, excl, s_i \mid k)$, nous recourons à une méthode exacte, à savoir : à la programmation linéaire en nombres entiers. Supposons qu'il y a k^0 stations, $k^0 \geq k^*$, qui peuvent être ouvertes ou pas ouvertes. Le nombre k^0 peut être obtenu par les heuristiques $GreedyNumberPar1$ et $GreedyNumberPar2$. Une station est appelée *ouverte* s'il y a au moins une opération qui lui est affectée.

Introduisons des 0-1 variables x_{ij} telles que $x_{ij} = 1$ si et seulement si l'opération i est affectée à la station j, et des 0-1 variables y_j telles que $y_j = 1$ si et seulement si la station j est ouverte. Rappelons que le graphe orienté $G = G(N, A)$ représente les contraintes de précédence. Calculons la borne inférieure $LBPar_1$ sur le nombre minimal de stations suivante : $LBPar_1 = \max\left\{l_{\max}, \left\lceil \frac{\sum_{i=1}^{n} s_i}{r} \right\rceil\right\}$, où $l_{\max}$ est le nombre des sommets dans la chaîne la plus longue de G. Donc, le problème $P_{par}(prec, excl, s_i \mid k)$ peut être reformulé comme le problème de programmation linéaire en nombres entiers suivant, noté comme $MinNumberPar$:

Problème $MinNumberPar$

$$\min \sum_{j=1}^{k^0} y_j, \tag{4.1}$$

sous les contraintes

$$\sum_{j=1}^{k^0} y_j \geq LBPar_1, \tag{4.2}$$

$$\sum_{i=1}^{j} y_i \geq jy_j, \quad j = 2, \ldots, k^0, \tag{4.3}$$

$$\sum_{i=1}^{n} x_{ij} \geq y_j, \quad j = 1, \ldots, k^0, \tag{4.4}$$

$$\sum_{j=1}^{k^0} x_{ij} = 1, \quad i = 1, \ldots, n, \tag{4.5}$$

$$\sum_{i=1}^{n} s_i x_{ij} \leq ry_j, \quad j = 1, \ldots, k^0, \tag{4.6}$$

$$\sum_{i \in E'} x_{ij} \leq |E'| - 1, \ \forall E' \in E, \ j = 1, \ldots, k^0, \tag{4.7}$$

$$\sum_{j=1}^{h} x_{bj} + \sum_{j=h}^{k^0} x_{aj} \leq 1, \ \forall (a,b) \in A, \ h = 1, \ldots, k^0, \tag{4.8}$$

$$x_{ij}, y_j \in \{0,1\}, \quad i = 1, \ldots, n, \ j = 1, \ldots, k^0. \tag{4.9}$$

Dans le problème $MinNumberPar$, il y a $k^0 + nk^0$ variables et $1 + n + k^0(3 + |E| + |A|)$ contraintes sans les 0-1 contraintes.

La contrainte (4.2) incorpore la borne inférieure $LBPar_1$. Les contraintes (4.3) garantissent que les stations sont ouvertes successivement dans l'ordre croissant de leurs indices et (4.4) qu'au moins une opération est affectée à une station ouverte. Notons ques les contraintes (4.2)-(4.4) ne sont pas nécessaires mais elles accélèrent le processus de résolution lors de l'utilisation d'un solveur. Les contraintes (4.5) assurent que chaque opération est affectée à une station. Les contraintes (4.6) garantissent que la capacité de chaque station n'est pas dépassée et que toute opération ne peut être affectée qu'à une station ouverte. Les contraintes (4.7) empêchent les opérations du même ensemble d'exclusion d'être entièrement affectées à la même station. Les contraintes (4.8) avec les contraintes (4.5) assurent que l'indice de station a est strictement inférieur à celui de b pour $(a,b) \in A$, ce qui représente les relations de précédence.

Le nombre minimal de stations peut être calculé comme $k^* = \sum_{j=1}^{k^0} y_j^*$, où les y_j^* sont les valeurs optimales y_j dans le problème $MinNumberPar$.

4.6 Minimisation du coût total de changements de séries

Dans la section précédente nous avons analysé le premier critère du problème étudié. Le problème correspondant $P_{par}(prec, excl, s_i \mid k)$ a été modélisé comme un problème de programmation linéaire en nombres entiers, dénommé $MinNumberPar$. Il a pour but de trouver le nombre minimal de stations. Maintenant, nous allons considérer le problème général d'optimisation lexicographique à deux critères $P_{par}(prec, excl, s_i \mid k, cost)$. D'abord, nous allons établir sa complexité de calcul. Notons que tout cas spécial du problème $P_{par}(prec, excl, s_i \mid k, cost)$ n'est pas plus facile que le même cas spécial du problème $P_{par}(prec, excl, s_i \mid k)$. Donc, les résultats de complexité pour le problème $P_{par}(prec, excl, s_i \mid k)$ de la section précédente s'appliquent pour le problème $P_{par}(prec, excl, s_i \mid k, cost)$.

Un algorithme de résolution pour le problème avec des tailles d'opérations unitaires sans contraintes de précédence et d'exclusion, désigné comme $P(s_i = 1 \mid k, cost)$, a été présenté dans [Kovalev et al., 2012]. Le temps de calcul de cet algorithme ne dépend que du nombre f des types d'opérations. Donc, le problème $P_{par}(s_i = 1 \mid k, cost)$ peut être résolu en temps polynomial pour un nombre constant de types d'opérations. Maintenant, nous allons montrer que le problème $P_{par}(s_i = 1 \mid k, cost)$ est NP-difficile au sens fort si f est variable (une partie de l'instance du problème) même si tous les coefficients de coût sont unitaires.

Théorème 2 *Le problème $P_{par}(s_i = 1 \mid k, cost)$ est NP-difficile au sens fort même si $a_v = 1$, $v = 1, \ldots, f$.*

Preuve: Nous utilisons une réduction du problème NP-difficile *3-Partition*, voir [Garey and Johnson, 1979] : en ayant $3q + 1$ nombres entiers positifs $b_1, \ldots, b_{3q}$ et B tels que $B/4 < b_i < B/2$, $i = 1, \ldots, 3q$, et $\sum_{i=1}^{3q} b_i = qB$, y a-t-il une partition de l'ensemble $\{1, \ldots, 3q\}$ en q ensembles disjoints $X_1, \ldots, X_q$ telle que $\sum_{i \in X_l} b_i = B$ pour $l = 1, \ldots, q$?

En ayant une instance de *3-Partition*, construisons l'instance suivante du problème $P_{par}(s_i = 1 \mid k, cost)$. Définissons le nombre d'opérations $n = qB$ et la capacité des stations $r = B$. Il y a $f = 3q$ types d'opérations. Chaque opération a un seul type et une taille unitaire. Il y a b_i opérations du type i, $i = 1, \ldots, 3q$. Donc, il y a qB opérations au total. Nous allons montrer que pour l'instance construite, une solution de *3-Partition* existe *si et seulement si* il

existe une solution admissible de l'instance construite du problème $P_{par}(s_i = 1 \mid k, cost)$ avec q stations et un coût total de changements de séries égal à $3q$. Il est évident que la construction est pseudo-polynomial.

Partie « si ». Supposons qu'il existe une solution admissible de l'instance construite du problème $P_{par}(s_i = 1 \mid k, cost)$ avec au plus $3q$ changements de séries. Il est facile de voir que le nombre minimal de stations est égal à q. Vu qu'il y a $3q$ types d'opérations, les opérations du même type doivent être affectées à la même station. Soit X_h l'ensemble des types des opérations affectées à la station h. Nous devons avoir $\sum_{i \in X_h} b_i \leq r = B$, $h = 1, \ldots, q$. Cela, avec l'égalité $\sum_{h=1}^{q} \sum_{i \in X_h} b_i = \sum_{i=1}^{3q} b_i = qB$, implique $\sum_{i \in X_h} b_i = B$, $h = 1, \ldots, q$, ce qui accomplit la preuve de la partie « si ».

Partie « seulement si ». S'il existe une solution $X_1, \ldots, X_q$ du problème *3-Partition*, alors construisons une solution du problème $P_{par}(s_i = 1 \mid k, cost)$ dans laquelle toutes les opérations de l'ensemble X_h sont affectées à la station h, $h = 1, \ldots, q$. Cette solution a q stations et un coût total $3q$, ce qui accomplit la preuve de la partie « seulement si ». ∎

4.6.1 Heuristique pour le problème $P_{par}(prec, excl, s_i \mid k, cost)$

En raison de la complexité du problème étudié, nous proposons une heuristique gloutonne randomisée, dénommé $GreedyCostPar$, pour résoudre le problème $P_{par}(prec, excl, s_i \mid k, cost)$ de manière approchée. Soit c_i le coût total de changements de séries de l'opération $i \in N$: $c_i = \sum_{v \in T_i} a_v$. La seule différence entre les heuristiques $GreedyCostPar$ et $GreedyNumberPar1$ est que dans $GreedyCostPar$, au lieu de résoudre le problème de la somme de sous-ensembles, le problème suivant du sac à dos est résolu :

$$\max_{X \in N^+} \sum_{i \in X} c_i, \text{ sous la contrainte } \sum_{i \in X} s_i \leq r.$$

Une solution optimale de ce problème affecte les opérations à coût maximal à la même station.

De nouveau, nous utilisons l'algorithme de programmation dynamique standard pour résoudre le problème du sac à dos. Cela nécessite un temps en $O(|N^+|r)$. L'heuristique $GreedyCostPar$ peut être appliquée plusieurs fois afin d'augmenter la probabilité d'obtenir une solution admissible à coût faible.

Notons que l'heuristique $GreedyCostPar$ peut également être utilisée pour résoudre le problème $P_{par}(prec, excl, s_i \mid k)$ de manière approchée. Aussi, les heuristiques $GreedyNumberPar1$ et $GreedyNumberPar2$ peuvent être utilisées afin de résoudre le problème $P_{par}(prec, excl, s_i \mid k, cost)$ de manière approchée.

4.6.2 Formulation en un programme linéaire en nombres entiers du problème $P_{par}(prec, excl, s_i \mid k, cost)$

Afin de résoudre le problème général $P_{par}(prec, excl, s_i \mid k, cost)$, nous recourons de nouveau à la programmation linéaire en nombres entiers. Supposons que le nombre minimal de stations k^* a été trouvé. Introduisons les 0-1 variables z_{vj} telles que $z_{vj} = 1$ si et seulement si au moins une opération de type v est affectée à la station j. Comme dans le problème $MinNumberPar$, les x_{ij} sont des 0-1 variables telles que $x_{ij} = 1$ si et seulement si l'opération i est affectée à la station j.

Calculons la borne inférieure suivante $LBPar_2$ sur le coût total minimal : $LBPar_2 = \sum_{v=1}^{f} a_v \max\left\{l_{\max}^{(v)}, \left\lceil \frac{\sum_{i \in N_v} s_i}{r} \right\rceil \right\}$, où $l_{\max}^{(v)}$ est le nombre maximal d'opérations du type v dans n'importe quelle chaîne de G. Le problème $P_{par}(prec, excl, s_i \mid k, cost)$ peut être reformulé comme le problème de programmation linéaire en nombres entiers suivant, noté comme $MinCostPar$:

Problème *MinCostPar*

$$\min \quad \sum_{j=1}^{k^*} \sum_{v=1}^{f} a_v z_{vj}, \tag{4.10}$$

sous les contraintes

$$\sum_{j=1}^{k^*}\sum_{v=1}^{f} a_v z_{vj} \geq LBPar_2, \tag{4.11}$$

$$\sum_{i\in N_v} x_{ij} \geq z_{vj}, \quad v=1,\ldots,f,\ j=1,\ldots,k^*, \tag{4.12}$$

$$\sum_{i=1}^{n} s_i x_{ij} \leq r, \quad j=1,\ldots,k^*, \tag{4.13}$$

$$\sum_{i\in N_v} x_{ij} \leq |N_v| z_{vj},\ v=1,\ldots,f,\ j=1,\ldots,k^*, \tag{4.14}$$

$$\sum_{j=1}^{k^*} x_{ij} = 1, \quad i=1,\ldots,n, \tag{4.15}$$

$$\sum_{i\in E'} x_{ij} \leq |E'|-1,\ \forall E'\in E,\ j=1,\ldots,k^*, \tag{4.16}$$

$$\sum_{j=1}^{h} x_{bj} + \sum_{j=h}^{k^*} x_{aj} \leq 1,\ \forall (a,b)\in A,\ h=1,\ldots,k^*, \tag{4.17}$$

$$x_{ij}, z_{vj} \in \{0,1\}, \quad i=1,\ldots,n,\ j=1,\ldots,k^*,\ v=1,\ldots,f. \tag{4.18}$$

Dans le problème $MinCostPar$ il y a $k^*(f+n)$ variables et $1+n+k^*(1+2f+|E|+|A|)$ contraintes sans les 0-1 contraintes.

La contrainte (4.11) incorpore la borne inférieure $LBPar_2$. Les contraintes (4.12), même si elles sont dégénérées, parce que les variables z_{vj} seront mises à zéro dans une solution optimale si $x_{ij}=0$ pour tous les $i\in N_v$, sont inclues afin d'améliorer le processus de résolution par le solveur. Les contraintes (4.13) concernent la capacité des stations. Les contraintes (4.14) garantissent que toute opération d'un certain type peut être affectée à une station si cette station est ouverte pour ce type. Les contraintes (4.15) - (4.17) ont la même signification que dans le problème $MinNumberPar$.

4.7 Résultats numériques

Nous avons utilisé le solveur IBM ILOG Cplex version 12.4. pour résoudre les problèmes $MinNumberPar$ et $MinCostPar$. Les expérimentations ont été effectuées sur un ordinateur avec un processeur Intel(R) Core(TM) Quad 2.83 GHz et une mémoire à accès direct de 8 Go sous le système d'exploitation MS Windows 7 Professional (64 bit). L'algorithme de génération des données et les heuristiques ont été codés en C++.

Dans nos expérimentations, la limite du temps de résolution de chaque instance du

$MinNumberPar$ ou $MinCostPar$ a été fixée à une heure.

Trois familles d'instances ont été aléatoirement générées. Une famille est associée à un nombre d'opérations $n \in \{40, 60, 80\}$ donné. Pour toutes les instances, il y a trois types de produits. Les coûts de changements de séries sont $a_1 = 3$, $a_2 = 2$ et $a_3 = 1$. La capacité des stations est égale à $r = 10$.

Une taille $s_i \in \{1, 2, 3\}$ et un ensemble de types $T_i \in \Big\{\{1,2,3\},\{1,2\},\{1,3\},\{2,3\},\{1\},\{2\},\{3\}\Big\}$ avec des probabilités également distribuées sont arbitrairement affectés à chaque opération $i \in N$. Les ensembles d'exclusions sont définis comme suit. Pour chaque paire (a, b) d'opérations telles qu'elles ont au moins un type commun, il est décidé avec la probabilité $1/3$ que a exclut b. Ainsi, tous les ensembles d'exclusion ont une cardinalité de deux.

Chaque famille est divisée en quatre *séries* n_1, n_2, n_3 et n_4 avec une densité de contraintes de précédence faible, moyenne, élevée et très élevée, respectivement. Pour chaque série n_e, $e \in \{1, 2, 3, 4\}$, et chaque paire (a, b) d'opérations telles que $a < b$ et a et b ont au moins un type commun, il est décidé avec la probabilité p_e que a précède b. Les probabilités correspondantes sont $p\_1 = 0,2$, $p\_2 = 0,4$, $p\_3 = 0,6$ et $p\_4 = 0,8$. Vu que chaque arc du graphe de précédence obtenu passe toujours d'un sommet avec un indice plus petit à un sommet avec un indice plus grand, ce graphe est acyclique.

Chaque série se compose de 20 instances. Leurs caractéristiques moyennes sont données dans le Tableau 4.7. $\%Prec$ représente la densité des contraintes de précédence, définies comme $\frac{|A|}{\max \#\text{ arcs}} \times 100\% = \frac{2|A|}{n^2-n} \times 100\%$.

Tableau 4.3 – Caractéristiques numériques moyennes

Séries n_e	40_1	40_2	40_3	40_4	60_1	60_2	60_3	60_4	80_1	80_2	80_3	80_4
$\%Prec$	9,35%	19,25%	29,74%	40,25%	9,88%	19,74%	29,74%	41,24%	10,09%	19,93%	30,09%	39,43%
$\sum_{i=1}^{n} s_i$	78	79	79	80	117	120	120	120	160	160	158	159
$\|E\|$	189	186	202	199	444	451	443	458	797	790	791	786

Pour résoudre chaque instance du problème $P_{par}(prec, excl, s_i \mid k, cost)$, la procédure suivante est utilisée.

Procédure de résolution pour le problème $P_{par}(prec, excl, s_i \mid k, cost)$:

1. Appliquer les heuristiques $GreedyNumberPar1$, $GreedyNumberPar2$ et $GreedyCostPar$. Déterminer le nombre minimal de stations, k^0, obtenu par ces heuristiques.
2. Appliquer Cplex pour résoudre le problème $MinNumberPar$. Déterminer le nombre minimal de stations k^*.
3. Appliquer Cplex pour résoudre le problème $MinCostPar$ avec le nombre de stations k^*.
4. Sélectionner la meilleure solution du problème $P_{par}(prec, excl, s_i \mid k^*, cost)$ parmi les solutions obtenues aux étapes 1, 2 et 3.

Les résultats possibles de chaque lancement de Cplex pour une instance I sont les suivants :

- Une solution optimale avec la valeur *Opt(I)* est trouvée
- Une solution approchée avec la valeur *Appr(I)*, une borne inférieure *LB(I)* et une erreur relative *%Gap(I)* sont obtenues.
- Aucune solution admissible n'a été trouvée lorsque la limite de temps a été atteinte.

Une erreur relative $\%Gap(I)$ est définie comme $\frac{Appr(I)-LB(I)}{LB(I)} \times 100\%$. Dans nos expérimentations le troisième cas n'a jamais eu lieu.

Le Tableau 4.4 montre le nombre d'instances du problème $MinNumberPar$ résolues à l'optimalité (ligne « *Opt* ») et de manière approchée (ligne « *Appr* ») pour chaque série. En outre, les erreurs relatives moyennes (ligne « $\%Err.\ moy.$ ») et les pires erreurs relatives (ligne « $\%Err.\ pire$ ») sont indiquées. L'erreur relative moyenne $\%Err.\ moy.$ est définie comme $\frac{1}{|n\_e|} \sum_{I \in n\_e} \%Gap(I)$ et l'erreur la pire $\%Err.\ pire$ comme $\max_{I \in n\_e} \{\%G(I)\}$ pour chaque série n_e, $e \in \{1, 2, 3, 4\}$. Pour presque toutes les instances la solution optimale a été trouvée. Seules 4 instances sur 240 ont été résolues approximativement avec une erreur relative acceptable. Quant au problème $MinCostPar$, *toutes les instances ont été résolues à l'optimalité pour toutes les séries.*

Les valeurs des fonctions objectifs peuvent être utilisées pour calculer l'erreur relative de chaque instance et déterminer les erreurs relatives moyennes et les pires erreurs relatives de chaque série. Les résultats des heuristiques *GreedyNumberPar1*, *GreedyNumberPar2* et *GreedyCostPar* sont présentés dans le Tableau 4.5. Les lignes *GreedyNumberPar* et *GreedyCostPar*

TABLEAU 4.4 – $MinNumberPar$: qualité des solutions

Séries *n_e*	40_1	40_2	40_3	40_4	60_1	60_2	60_3	60_4	80_1	80_2	80_3	80_4
Opt	20	20	20	20	20	20	20	20	17	19	20	20
Appr	0	0	0	0	0	0	0	0	3	1	0	0
%Err. moy.	-	-	-	-	-	-	-	-	9,10	16,67	-	-
%Err. pire	-	-	-	-	-	-	-	-	13,1	16,67	-	-

contiennent les meilleurs résultats des deux heuristiques *GreedyNumberPar1* et *GreedyNumberPar2* au regard du nombre minimal de stations, et de l'heuristique *GreedyCostPar* concernant coût minimal total de changements de séries.

Deux conclusions peuvent être tirées de ces résultats. En premier lieu, les heuristiques se trouvent en moyenne dans un intervalle entre 10% et 45% de l'optimum, sachant que Cplex a trouvé les solutions optimales pour pratiquement chaque instance. En deuxième lieu, il y a une corrélation forte entre la densité des contraintes de précédence et la qualité des solutions obtenues avec les heuristiques. Plus la densité des contraintes de précédence est élevée, meilleures sont les solutions. Par ailleurs, même si les heuristiques donnent des résultats avec des erreurs parfois importantes, elles sont nécessaires pour l'obtention du paramètre k^0 - la borne supérieure sur le nombre de stations employée dans $MinNumberPar$.

TABLEAU 4.5 – Résultats des heuristiques *GreedyNumberPar* et *GreedyCostPar*

Séries *n_e*	40_1	40_2	40_3	40_4	60_1	60_2	60_3	60_4	80_1	80_2	80_3	80_4
GreedyNumberPar %Err. moy.	42,63	21,42	20,98	11,09	32,99	23,26	19,33	11,32	31,79	22,47	16,29	9,74
GreedyNumberPar %Err. pire	66,67	38,53	37,57	25,52	46,15	39,83	32,78	20,90	50,12	39,35	31,67	24,16
GreedyCostPar %Err. moy.	32,73	26,13	17,86	9,64	26,49	21,78	16,78	11,32	29,74	24,48	22,28	9,31
GreedyCostPar %Err. pire	54,60	41,18	34,05	17,78	42,96	35,86	29,69	23,14	53,87	41,33	32,35	23,60

Le Tableau 4.6 montre les temps de calcul moyens (ligne « *Temps moy.* ») et maximaux (ligne « *Temps max.* ») pour la résolution des problèmes *MinNumberPar* et *MinCostPar* par Cplex. On peut faire plusieurs observations. Premièrement, les résultats prouvent que Cplex résout des instances du problème $P_{par}(prec, excl, s_i \mid k, cost)$ en un temps court. En moyenne, les instances sont résolues en moins de 25 minutes. Deuxièmement, nous pouvons constater que le problème *MinNumerPar* exige plus de temps pour la résolution que *MinCostPar*. Cela peut être lié aux valeurs élevées du paramètre k^0 fournies par les heuristiques. La troisième

observation concerne la densité des contraintes de précédence. Normalement, les instances avec une densité faible nécessitent un effort de calcul plus élevé.

Tableau 4.6 – Temps de calcul du Cplex pour la résolution des problèmes *MinNumberPar* et *MinCostPar*

Séries n_e	40_1	40_2	40_3	40_4	60_1	60_2	60_3	60_4	80_1	80_2	80_3	80_4
Temps moy. MinNumberPar (sec)	5,99	4,91	3,06	2,08	151,80	94,58	27,20	26,61	1416,60	1144,04	718,22	249,84
Temps max. MinNumberPar (sec)	18,11	13,00	12,18	7,02	312,19	280,97	68,95	173,29	3600	3600	3137,18	1312,09
Temps moy. MinCostPar sec)	0,84	0,30	0,37	0,82	5,08	1,41	1,96	5,83	145,56	65,45	9,43	38,86
Temps max. MinCostPar (sec)	2,31	0,86	0,64	2,11	17,96	3,26	3,9	28,77	1037,45	1182,77	18,63	309,05

Nous déduisons que Cplex peut être utilisé pour résoudre des instances de dimension pratique en un temps acceptable. Notons que dans le chapitre 3 nous avons étudié un problème de configuration d'une ligne de fabrication avec des instances dont la taille allait jusqu'à 130 opérations. Ces opérations étaient unitaires, c'est-à-dire que leur taille était égal à un. Vu que la taille moyenne des opérations que nous utilisons ici pour le problème $P_{par}(prec, excl, s_i \mid k, cost)$ est égale à deux, les instances les plus larges correspondent donc à environ 160 opérations unitaires.

4.8 Conclusion

Le problème de configuration d'une ligne de fabrication multi-produit cadencée a été étudié. Le cheminement de production de cette ligne est unique pour tous les types de produits, et son schéma logique est linéaire. Chaque pièce de produit d'un certain type est transportée entre les stations dans la même direction. Un ensemble d'opérations spécifiques est associé à chaque type de pièces. Toutes les opérations de tous les types doivent être affectées aux stations. Les opérations affectées à une station sont exécutées en parallèle. Un changement de séries a lieu sur une station si au moins une opération d'un certain type est affectée à cette station. Chaque changement de séries implique les démarches nécessaires pour commencer et finir le traitement du produit correspondant sur la station. Un coût de changement de séries est associé à chaque type de produits. Chaque opération sous-entend l'utilisation d'un ou de plusieurs outils. Le nombre d'outils nécessaires pour effectuer une opération est appelé une « taille d'opération ».

La taille totale des opérations sur une station est limitée. Des contraintes de précédence et d'exclusion sont données sur l'ensemble des opérations. Nous supposons qu'il y a une borne supérieure commune sur les temps de cycle de tous les types de produits et qu'elle n'est pas dépassée. Le problème a deux critères considérés lexicographiquement : la minimisation du nombre des stations et la minimisation du coût total de changements de séries.

Nous avons établi la complexité de calcul pour plusieurs cas de ce problème. Les résultats de l'analyse de complexité sont donnés dans le Tableau 4.7.

TABLEAU 4.7 – Complexité de calcul pour les cas spéciaux du problème $P_{par}(prec, excl, s_i \mid k, cost)$

Problème	Complexité
$P_{par}(prec, s_i = 1 \mid k)$, r arbitraire	NP-difficile au sens fort
$P_{par}(prec, s_i = 1 \mid k)$, r constant	ouvert
$P_{par}(in - tree, s_i = 1 \mid k)$	$O(n)$
$P(out - tree, s_i = 1 \mid k)$	$O(n)$
$P_{par}(prec, s_i = 1 \mid k)$, $r = 2$	$O(n^2)$
$P_{par}(s_i \mid k)$, r arbitraire	NP-difficile au sens fort
$P_{par}(s_i \mid k)$, r constant	$O(n^{const})$
$P_{par}(excl, s_i = 1 \mid k)$	NP-difficile au sens fort
$P_{par}(s_i = 1 \mid k, cost)$, f arbitraire	NP-difficile au sens fort
$P_{par}(s_i = 1 \mid k, cost)$, f constant	$O(n^{const})$

La procédure de résolution du problème combine trois heuristiques gloutonnes et un programme linéaire en nombres entiers qui est résolu à l'aide du solveur Cplex. Les expérimentations numériques ont montré que notre approche est capable de résoudre des instances de dimension pratique en un temps acceptable.

Chapitre 5

Minimisation des coûts de changements de séries pour des lignes multi-produits : cas d'exécution séquentielle des opérations

5.1 Introduction

Contrairement au problème examiné dans le chapitre 4, ici, nous allons étudier un problème avec un mode d'exécution séquentiel des opérations. Cela suscite quelques changements importants dans la structure du problème et nous allons les indiquer en rappelant cependant les notations introduites dans le chapitre 4. Comme avant, nous nous appuyons sur la classification à six éléments du chapitre 2.

5.2 Définition du problème

La **variabilité de ligne**. Une ligne doit être configurée pour la production de masse de f types de produits. La ligne est donc *flexible* ou *reconfigurable*.

Soit $F = \{1, \ldots, f\}$ l'ensemble des types de produits. Chaque produit de type $v \in F$ nécessite l'exécution une seule fois sur la ligne d'un ensemble donné N_v d'opérations. Les opérations de l'ensemble N_v sont appelées des *opérations de type* v. Différents ensembles N_v peuvent contenir des opérations communes. Ainsi, la même opération peut être de différents types. Soit $N := \cup_{v=1}^{f} N_v = \{1, \ldots, n\}$ le surensemble de toutes les opérations nécessaires, et soit T_i

l'ensemble des types de l'opération $i \in N$, c'est-à-dire, $T_i = \{v \mid i \in N_v,\ v \in F\}$. Comme dans le chapitre 4, la réaffectation d'opérations (rééquilibrage) n'est pas permise.

Les **caractéristiques de ligne**. La ligne est cadencée et le *cheminement de production est unique* car il n'y a pas de stations parallèles. Le *schéma logique de la ligne est linéaire* : les produits de tous les types sont transportés entre des stations dans la même direction.

Les **caractéristiques de temps**. Les *temps opératoires sont déterministes*. À la différence du chapitre 4, nous devons ici tenir compte des notions de temps de changement de séries et de temps opératoire. Un temps de changement de séries t_v est associé à un ensemble d'opérations de type v affectées à la même station, $v = 1, \ldots, f$. Un temps opératoire p_i est associé à chaque opération $i \in N$.

Contrairement au problème présenté dans le chapitre 4 où l'ordre des produits de différents types peut être arbitraire sans violation de la borne supérieure commune sur les temps de cycle, ici les produits de tous les types sont lancés sur la ligne par lot selon le principe de la *technologie de groupe*. C'est-à-dire les produits d'un certain type entrent tous dans la ligne l'un après l'autre et qu'il n'y a pas de produit des autres types entre eux. La fabrication par lot est largement utilisée dans les systèmes de production, voir, par exemple, [Tatikonda and Wemmerlöv, 1992, Radharamanan, 1994, Gunasekaran et al., 2001]. Compte tenu de cette fabrication par lot de produits homogènes, nous pouvons déterminer le temps de cycle comme le temps entre les sorties de deux produits consécutifs qui est égal au temps de station maximal.

Les **caractéristiques de coût**. S'il y a au moins une opération d'un certain type affectée à une station, alors le changement de séries aura lieu sur cette station. Un coût de changement de séries a_v est associé au type de produits v, $v = 1, \ldots, f$.

Les **contraintes technologiques**. Dans ce problème, le *mode d'exécution des opérations* sur les stations est **séquentiel** : des opération du même type affectées à la même station sont exécutées successivement.

Les *contraintes de précédence* sont données sur le surensemble N. Si une opération $i \in N$ précède une opération $j \in N$, alors les opérations i et j sont de même type, l'opération j ne peut pas être affectée à une station qui précède la station de i, et si l'opération j est affectée à la station de i, elle ne peut pas être effectuée avant i. Les contraintes de précédence sont

représentées par un graphe orienté acyclique $G = G(N, A)$, dans lequel il y a un arc $(i, j) \in A$ si et seulement si une opération i précède une opération j. Notons que, à la différence du chapitre 4, les opérations affectées à la même station peuvent toujours être ordonnées à l'égard des contraintes de précédence, voir, par exemple, la procédure du tri topologique de [Kahn, 1962].

Chaque opération i a sa *taille* s_i qui représente le nombre d'outils standard nécessaires pour effectuer cette opération. La taille totale de toutes les opérations affectées à la même station (le nombre d'outils standard) ne doit pas dépasser la *capacité de station* r.

Les temps opératoires de toutes les opérations du même type v affectées à la même station plus le temps de changement de séries correspondant ne doivent pas dépasser la borne supérieure U_v, $v = 1, \ldots, f$. Autrement dit, la somme des temps opératoires des opérations de type v ne doit pas dépasser $U_v - t_v$. Donc, nous pouvons simplifier la formulation du problème en mettant $U_v := U_v - t_v$, $v \in F$. Donc, U_v représente une *borne supérieure sur le temps de cycle du produit de type* v.

Les **critères d'optimisation**. Soit x_v le nombre de changements de séries pour un produit de type v, $v = 1, \ldots, f$. Une décision doit être prise à propos du nombre de stations, k, et de l'affectation des opérations aux stations $1, \ldots, k$ afin que la capacité des stations r et les bornes supérieures sur les temps de cycle $U_1, \ldots, U_f$ soient vérifiées.

Le problème a deux critères identiques à ceux du chapitre 4. Le critère principal est la *minimisation du nombre de stations*, k. Le critère secondaire est la *minimisation du coût total de changements de séries*, $a_1 x_1 + \cdots + a_f x_f$.

Nous désignons le problème avec seulement le premier critère comme $P_{seq}(prec \mid k)$ et le problème avec deux critères ordonnés comme $P_{seq}(prec \mid k, cost)$. Le problème $P_{seq}(prec \mid k, cost)$ consiste à minimiser le coût total de changements de séries sur l'ensemble des solutions admissibles pour le problème $P_{seq}(prec \mid k)$. Soit (k^*, c^*) les valeurs optimales pour le problème $P_{seq}(prec \mid k, cost)$, où k^* est le nombre minimal des stations et c^* est le coût total de changements de séries.

Afin d'éliminer des instances inadmissibles, nous supposons, sans perte de généralité, que $s_i \leq r$ et $p_i \leq U_v$ pour toutes les opérations et tous les types. Avec ces suppositions, une solution admissible pour le problème $P_{seq}(prec \mid k, cost)$ existe toujours.

5.3 Origine du problème

Une application de ce problème dans l'industrie est identique à celle décrite dans le chapitre 4, sauf quelques particularités importantes, à savoir :

- Une ligne est destinée à la production de masse de plusieurs types de produits selon le *principe de la technologie de groupe*.
- Lorsqu'un produit d'un nouveau type entre dans une station, un changement de séries est effectué. Les outils inutiles sont enlevés du boîtier multibroche et sont retournés au magasin. Ensuite, les outils nécessaires sont *successivement* pris du magasin et sont montés sur le boîtier. Après l'exécution de l'opération correspondante, ils sont démontés et retournés au magasin.
- Le temps opératoire de chaque opération englobe l'intervalle de temps à partir du démontage des outils inutiles jusqu'à la fin d'opération. En nous fondant sur des observations, nous supposons raisonnablement que les *temps de démontage sont négligeables et que les temps de montage sont inclus dans les temps opératoires*. Lorsque toutes les opérations nécessaires sur une station sont terminées, un nettoyage de la station est effectué. Son temps (coût) est inclus dans le temps (coût) de changement de séries.

5.4 Exemple d'une solution

Une solution pour ce problème peut être représentée comme un tableau dont les colonnes représentent des stations et un rectangle dans la colonne représente une opération i suivie par ses types (ensemble T_i) entre parenthèses. Une opération d'un rectangle supérieur est supposée être effectuée plus tôt que l'opération d'un rectangle inférieur. La taille d'une opération est le nombre de lignes dans le rectangle. Considérons un problème avec la capacité des stations $r = 6$, 16 opérations de trois types et sans contraintes de précédence. Le Tableau 5.1 représente une solution pour ce problème avec un nombre minimal de stations $k^* = 4$ et un coût total de changements de séries $3a_1 + 3a_2 + 3a_3$ parce que chaque type est présent sur trois stations.

Notons que la minimisation du coût total de changements de séries n'est pas suffisante pour la minimisation du nombre de stations et inversement. Pour notre exemple, le coût total de

TABLEAU 5.1 – Exemple d'une solution à quatre stations, $cost = 3a_1 + 3a_2 + 3a_3$

<table>
<tr><th></th><th>St. 1</th><th>St. 2</th><th>St. 3</th><th>St. 4</th></tr>
<tr><td>1</td><td rowspan="3">2(1,2,3)</td><td>1(1,2)</td><td rowspan="2">3(1)</td><td>12(2)</td></tr>
<tr><td>2</td><td rowspan="3">4(1,2)</td><td>13(2)</td></tr>
<tr><td>3</td><td>5(1)</td><td>14(2)</td></tr>
<tr><td>4</td><td rowspan="2">7(1,2,3)</td><td>6(1)</td><td>15(2)</td></tr>
<tr><td>5</td><td rowspan="2">10(1,2)</td><td>8(3)</td><td rowspan="2">16(3)</td></tr>
<tr><td>6</td><td>9(2,3)</td><td>11(3)</td></tr>
</table>

changements de séries peut être réduit en augmentant le nombre de stations, voir le Tableau 5.2.

TABLEAU 5.2 – Exemple d'une solution à cinq stations, $cost = 3a_1 + 3a_2 + 2a_3$

<table>
<tr><th></th><th>St. 1</th><th>St. 2</th><th>St. 3</th><th>St. 4</th><th>St. 5</th></tr>
<tr><td>1</td><td rowspan="3">2(1,2,3)</td><td>1(1,2)</td><td rowspan="2">3(1)</td><td>12(2)</td><td>8(3)</td></tr>
<tr><td>2</td><td rowspan="3">4(1,2)</td><td>13(2)</td><td>11(3)</td></tr>
<tr><td>3</td><td>5(1)</td><td>14(2)</td><td rowspan="2">16(3)</td></tr>
<tr><td>4</td><td rowspan="2">7(1,2,3)</td><td>6(1)</td><td>15(2)</td></tr>
<tr><td>5</td><td rowspan="2">10(1,2)</td><td rowspan="2">-</td><td rowspan="2">-</td><td rowspan="2">-</td></tr>
<tr><td>6</td><td>9(2,3)</td></tr>
</table>

5.5 Minimisation du nombre de stations

Dans cette section nous étudions le problème à un seul critère $P_{seq}(prec \mid k)$ qui implique la minimisation du nombre de stations. Si les temps opératoires ou les tailles des opérations sont arbitraires, alors le problème de décision concernant l'existence d'une solution admissible au regard des contraintes sur le temps de cycle ou sur la capacité des stations est NP-difficile au sens fort même s'il y a un seul type de produits et qu'il n'y a pas de contraintes de précédence. En effet, ce problème contient le problème de *3-Partition* comme un cas particulier avec des temps opératoires p_i ou des tailles s_i qui jouent le rôle de valeurs des nombres dans *3-Partition*. Rappelons que le problème de *3-Partition* est NP-difficile au sens fort.

Dans la sous-section 5.5.1, nous allons considérer le problème avec des temps opératoires unitaires et des tailles d'opérations unitaires, $P_{seq}(p_i = 1, s_i = 1, prec \mid k)$. Ce problème modélise une situation fréquente dans l'usinage des métaux lorsqu'une opération d'usinage est associée à un trou et les temps opératoires sont pratiquement identiques. Ce problème a des cas NP-difficiles ainsi que des cas pouvant être résolus en temps polynomial.

Dans les sous-sections 5.5.2 et 5.5.3 nous proposons deux heuristiques constructives et une formulation de programmation linéaire en nombres entiers, respectivement, pour le cas général du problème $P_{seq}(prec \mid k)$.

5.5.1 Temps opératoires et tailles d'opérations unitaires

D'abord, nous allons prouver que le problème $P_{seq}(p_i = 1, s_i = 1, prec \mid k)$ est NP-difficile au sens fort même pour le cas de trois types de pièces, de contraintes de précédence en forme de chaîne et sans contrainte sur la capacité des stations. Désignons ce cas comme $P_{seq}(f = 3, p_i = 1, s_i = 1, chains, r = \infty \mid k)$. Ensuite, nous allons montrer que le problème $P_{seq}(p_i = 1, s_i = 1, prec \mid k)$ peut être résolu en temps polynomial s'il n'y a pas de contrainte sur le temps de cycle.

Théorème 3 *Le problème $P_{seq}(f = 3, p_i = 1, s_i = 1, chains, r = \infty \mid k)$ est NP-difficile au sens fort.*

Preuve: Nous utilisons une réduction du problème *3-Partition* qui est NP-difficile au sens fort, voir [Garey and Johnson, 1979]. Le problème est le suivant. En ayant $3q + 1$ nombres entiers positifs $b_1, \ldots, b_{3q}$ et B tels que $B/4 < b_i < B/2$, $i = 1, \ldots, 3q$, et $\sum_{i=1}^{3q} b_i = qB$, existe-t-il une partition de l'ensemble $\{1, \ldots, 3q\}$ en q ensembles disjoints $X_1, \ldots, X_q$ tels que $\sum_{i \in X_h} b_i = B$ pour $h = 1, \ldots, q$?

En ayant une instance de *3-Partition*, construisons l'instance suivante de la version de décision du problème $P_{seq}(f = 3, p_i = 1, s_i = 1, chains, r = \infty \mid k)$. Définissons le nombre d'opérations $n = 6q + qB$ et la capacité de stations $r = \infty$. Il y a $f = 3$ types d'opérations. Les opérations $1, \ldots, qB$ sont des opérations du type 1, les opérations $qB + 1, qB + 2, \ldots, qB + 3q$ sont du type 1 et 2 en même temps, et les opérations $qB + 3q + 1, qB + 3q + 2, \ldots, qB + 6q$ sont les

opération du type 1 et 3 en même temps. La borne supérieure sur le temps de cycle pour les opération du type 1 est égale à $U_1 = B + 6$, et les bornes supérieures sur le temps de cycle pour les opérations du type 2 et 3 sont égales à trois, $U_2 = U_3 = 3$. Le graphe de précédence est une collection de $3q$ chaînes $C_1, \ldots, C_{3q}$ telles qu'une chaîne C_h commence avec une opération $qB + h$ du type 1 et 2, ensuite, b_h opérations arbitraires du type 1 suivent, et elle se finit par une opération $qB + 3q + h$ du type 1 et 3, $h = 1, \ldots, 3q$.

Nous allons montrer que pour l'instance ainsi construite, une solution de *3-Partition* existe *si et seulement si* il existe une solution admissible pour cette instance du problème $P_{seq}(f = 3, p_i = 1, s_i = 1, chains, r = \infty \mid k)$ avec un nombre de stations inférieur à q. Il est évident que la construction d'instance est pseudo-polynomial.

Partie « si ». Supposons qu'il existe une solution admissible pour l'instance construite du problème $P_{seq}(f = 3, p_i = 1, s_i = 1, chains, r = \infty \mid k)$ avec au plus q stations. Puisque la borne supérieure sur le temps de cycle pour le type 3 est égal à 3, au plus trois opérations du type 3 peuvent être affectées à la même station. De plus, puisqu'il y a au plus q stations et $3q$ opérations du type 3, il doit y avoir exactement q stations et exactement trois opérations du type 3 sur chaque station. Sinon, au moins quatre opérations du type 3 seraient affectées à au moins une station et donc, la limite sur le temps de cycle serait violée. Le même argument montre qu'exactement trois opérations du type 2 sont affectées à chaque station.

Considérons la station 1. Afin d'affecter de façon admissible trois opérations quelconques du type 3 à cette station, tous les prédécesseurs de ces trois opérations doivent être affectés à cette station. Donc, toutes les opérations de trois chaînes quelconques doivent être affectées à la station 1.

Considérons la station 2. De nouveau, pour affecter de façon admissible trois opérations quelconques du type 3 à cette station, tous les prédécesseurs de ces trois opérations doivent être affectés à cette station et à la station 1. Aucun prédécesseur de type 2 de ces opérations ne peut être affecté à la station 1 parce que sinon celle-ci contiendrait quatre opérations de type 2 et la limite sur le temps de cycle serait violée. Donc, toutes les opérations de trois chaînes quelconques doivent être affectées à la station 2. En suivant cette logique, nous déduisons que toutes les opérations de trois chaînes quelconques doivent être affectées à chacune des q stations.

Soit X_j l'ensemble des indices des chaînes affectées à une station j. Afin de vérifier la contrainte sur le temps de cycle pour des opérations du type 1, il doit y avoir $6 + \sum_{i \in X_j} b_i \leq U = B + 6$, $j = 1, \ldots, q$. Cela, avec l'égalité $\sum_{j=1}^{q} \sum_{i \in X_j} b_i = \sum_{i=1}^{3q} b_i = qB$, implique $\sum_{i \in X_j} b_i = B$,$j = 1, \ldots, q$, ce qui est nécessaire pour la preuve de la partie « si ».

Partie « seulement si ». S'il existe une solution $X_1, \ldots, X_q$ pour le problème *3-Partition*, alors construisons une solution pour le problème $P_{seq}(f = 3, p_i = 1, s_i = 1, chains, r = \infty \mid k)$ dans laquelle toutes les opérations des chaînes avec des indices de l'ensemble X_j sont affectées à la station j, $j = 1, \ldots, q$. Cette solution est admissible et a un nombre minimal de stations, q, ce qui est nécessaire pour la preuve de la partie « seulement si ». ■

Considérons le problème $P_{seq}(p_i = 1, s_i = 1, prec, U_v = \infty \mid k)$ avec des temps de cycle illimités. Notons que la condition $U_v = \infty$, $v \in F$, peut être remplacée par la condition $U_v = r, v \in F$. Cela signifie que les bornes supérieures sur les temps de cycle sont aussi grandes que la capacité des stations. Dans ce cas, les contraintes sur le temps de cycle sont redondantes avec les contraintes sur la capacité des stations et elles sont donc superflues.

Nous allons montrer que le problème $P_{seq}(p_i = 1, s_i = 1, prec, U_v = \infty \mid k)$ peut être résolu en temps $O(n + |A|)$ par une petite modification de la procédure d'ordonnancement topologique connue, voir, par exemple [Kahn, 1962]. Cette procédure crée un ordonnancement linéaire, désigné comme L, de sommets d'un graphe orienté acyclique tel que pour chaque arc (i, j) de ce graphe, i apparait avant j dans cet ordonnancement.

Ordonnancement topologique d'un graphe orienté acyclique $G = (N, A)$

1. Mettre $L = \phi$. Construire un ensemble N^+ des sommets qui n'ont aucun prédécesseur dans G.
2. Retirer un sommet arbitraire i de N^+. Insérer i dans L. Pour chaque sommet j qui est un successeur de i, retirer l'arc (i, j) du graphe. Si j de l'arc retiré (i, j) n'a aucun prédécesseur, alors insérer j dans N^+. Si N^+ est vide, alors arrêter : un ordre topologique L est construit. Sinon, répéter l'étape 2.

Soit L la sortie de la procédure d'ordonnancement topologique. Il est facile à voir qu'une solution optimale pour le problème $P_{seq}(p_i = 1, s_i = 1, prec, U_v = \infty \mid k)$ peut être obtenue par

une partition arbitraire de la séquence L en $k^* = \lceil \frac{n}{r} \rceil$ sous-séquences consécutives. Chaque sous-séquence inclue au plus r opérations, telles que $L = (L_1, \ldots, L_{k^*})$. Dans la solution optimale pour le problème $P_{seq}(p_i = 1, s_i = 1, prec, U_v = \infty \mid k)$, une station h effectue les opérations de la sous-séquence L_h dans son ordre, $h = 1, \ldots, k^*$.

Considérons le problème $P_{seq}(p_i = 1, s_i = 1, U_v = \infty \mid k)$ sans contrainte de précédence. Dans ce problème, les contraintes sur la capacité des stations sont les seules contraintes qui restreignent l'affectation des opérations aux stations. Ce problème peut être résolu en temps $O(1)$ en créant $k^* = \lceil \frac{n}{r} \rceil$ stations et en affectant de façon arbitraire r opérations à chaque station avec une exception possible pour une station.

5.5.2 Heuristique pour le cas général du $P_{seq}(prec \mid k)$

Afin de résoudre approximativement le problème $P_{seq}(prec \mid k)$, nous proposons une heuristique constructive gloutonne $GreedyNumberSeq$ et une heuristique constructive randomisée $RandomNumberSeq$. Numérotons les stations $h = 1, 2, \ldots$

Heuristique $GreedyNumberSeq$ pour le problème $P_{seq}(prec \mid k)$

1. Mettre $h = 1$. Construire un ensemble N^+ des sommets qui n'ont aucun prédécesseur dans $G = (N, A)$.
2. Pour une station h, calculer une *réserve de capacité* $C = r$ et une *réserve de temps de cycle* $R_v = U_v$ pour chaque type $v = 1, \ldots, f$.
3. Sélectionner une opération $i \in N^+$ avec la taille maximale s_i telle que $s_i \leq C$ et $p_i \leq R_v$ pour tous les types $v \in T_i$. Si une telle opération n'existe pas, alors remettre $h := h + 1$ et passer à l'étape 2. Si une telle opération i est sélectionnée, alors effectuer les actions suivantes. Affecter cette opération à la station h, la retirer de N^+, et retirer les arcs (i, j) du graphe. Si j de l'arc retiré (i, j) n'a aucun prédécesseur, alors insérer j dans N^+. Si N^+ est vide, alors arrêter : une solution admissible avec h stations est construite. Si N^+ n'est pas vide, alors remettre $C := C - s_i$, $R_v := R_v - p_i$ pour tous $v \in T_i$ et répéter l'étape 3.

L'heuristique $GreedyNumberSeq$ peut être résolu en temps $O(nf + |A|)$.

L'heuristique randomisée $RandomNumberSeq$ se distingue de $GreedyNumberSeq$ en ce qu'au lieu de sélectionner une opération $i \in N^+$ avec la taille maximale s_i à l'étape 3, une opération $i \in N^+$ est sélectionnée de manière randomisée. L'heuristique $RandomNumberSeq$ peut être exécutée plusieurs fois pour augmenter la probabilité de la construction d'une solution admissible avec un nombre réduit de stations.

5.5.3 Formulation en un programme linéaire en nombres entiers du problème général $P_{seq}(prec \mid k)$

Appelons une station *ouverte* si au moins une opération lui est affectée. Pour un programme linéaire en nombres entiers, supposons qu'il y a k^0 stations, $k^0 \geq k^*$, qui doivent être ouvertes ou pas. Le nombre k^0 peut être obtenu par les heuristiques $GreedyNumberSeq$ et $RandomNumberSeq$. Introduisons des variables 0-1 x_{ij} telles que $x_{ij} = 1$ si et seulement si une opération i est affectée à une station j, et des variables 0-1 y_j telles que $y_j = 1$ si et seulement si une station j est ouverte. Rappelons que le graphe orienté $G = (N, A)$ représente les contraintes de précédence.

Calculons la borne inférieure $LBSeq_1$ sur le nombre minimal de stations :

$$LBSeq_1 = \max\left\{\left\lceil \frac{\sum_{i=1}^{n} s_i}{r} \right\rceil, \max_{1 \leq v \leq f}\left\{\left\lceil \frac{\sum_{i \in N_v} p_i}{U_v} \right\rceil\right\}\right\}.$$

Le problème $P_{seq}(prec \mid k)$ peut être reformulé comme le problème de programmation linéaire en nombres entiers suivant que nous dénotons $MinNumberSeq$:

Problème *MinNumberSeq*

$$\min \sum_{j=1}^{k^0} y_j, \tag{5.1}$$

sous les contraintes

$$\sum_{j=1}^{k^0} y_j \geq LBSeq_1, \tag{5.2}$$

$$\sum_{j=1}^{k^0} x_{ij} = 1, \quad i = 1, \ldots, n, \tag{5.3}$$

$$\sum_{i=1}^{n} s_i x_{ij} \leq r y_j, \quad j = 1, \ldots, k^0, \tag{5.4}$$

$$\sum_{i \in N_v} p_i x_{ij} \leq U_v y_j, \ v = 1, \ldots, f, \ j = 1, \ldots, k^0, \tag{5.5}$$

$$\sum_{j=1}^{h} x_{bj} + \sum_{j=h+1}^{k^0} x_{aj} \leq 1, \ \forall (a,b) \in A, \ h = 1, \ldots, k^0 - 1, \tag{5.6}$$

$$\sum_{i=1}^{j} y_i \geq j y_j, \quad j = 1, \ldots, k^0, \tag{5.7}$$

$$x_{ij}, y_j \in \{0,1\}, \quad i = 1, \ldots, n, \ j = 1, \ldots, k^0. \tag{5.8}$$

Dans le problème $MinNumberSeq$, il y a $k^0 + nk^0$ variables et $1 + n + k^0(2 + f + |A|) - |A|$ contraintes sans les contraintes de type.

La contrainte (5.2) incorpore la borne inférieure $LBSeq_1$. Les contraintes (5.3) garantissent que chaque opération est affectée à une station. Les contraintes (5.4) assurent que la capacité de chaque station n'est pas dépassée et que toute opération ne peut être affectée qu'à une station ouverte. Les contraintes (5.5) garantissent que les bornes supérieures sur le temps de cycle ne sont pas dépassées pour chaque type de produits et chaque station ouverte. Les contraintes (5.6) avec (5.3) assurent que l'indice d'une station a n'est pas plus large que celui d'une station b pour $(a, b) \in A$, ce qui représente l'exigence des contraintes de précédence. Les contraintes (5.7) garantissent que les stations sont ouvertes successivement dans l'ordre croissant de leur indice.

Le nombre minimal de stations peut être calculé comme $k^* = \sum_{j=1}^{k^0} y_j^*$, où les y_j^* sont les valeurs optimales des variables y_j dans le problème $MinNumberSeq$.

5.6 Minimisation du coût total de changements de séries

Dans cette section nous allons établir la complexité de calcul, présenter une heuristique constructive et une formulation de programmation linéaire en nombres entiers pour le problème $P_{seq}(prec \mid k, cost)$. Notons que tout cas particulier du problème $P_{seq}(prec \mid k, cost)$ n'est

pas plus facile que le cas particulier correspondant du problème $P_{seq}(prec \mid k)$. Donc, les résultats concernant la complexité pour le problème $P_{seq}(prec \mid k)$ s'appliquent pour le problème $P_{seq}(prec \mid k, cost)$.

Considérons le problème $P_{seq}(p_i = 1, s_i = 1, U_v = \infty \mid k, cost)$ sans contrainte de précédence et contrainte sur le temps de cycle. Notons que pour ce problème le mode d'exécution des opérations ne joue pas un rôle. Un algorithme de résolution pour tel problème avec le mode d'exécution parallèle des opérations a été présenté dans [Kovalev et al., 2012]. Le temps de résolution de cet algorithme dépend seulement du nombre de types d'opérations f. De plus, il a été prouvé que l'équivalent du problème $P_{seq}(p_i = 1, s_i = 1, U_v = \infty \mid k, cost)$ avec le mode d'exécution parallèle des opérations est NP-difficile au sens fort si f est arbitraire (une partie de l'instance du problème) même si tous les coefficients de coût sont unitaires. Ces résultats s'appliquent aussi au problème $P_{seq}(p_i = 1, s_i = 1, U_v = \infty \mid k, cost)$.

Afin de résoudre approximativement le problème $P_{seq}(prec \mid k, cost)$, nous proposons une heuristique gloutonne, désignée comme *GreedyCostSeq*. La seule différence entre l'heuristique *GreedyCostSeq* et *GreedyNumberSeq* est que dans *GreedyCostSeq*, au lieu de sélectionner une opération $i \in N^+$ avec la taille maximale s_i à l'étape 3, une opération $i \in N^+$ avec le coût total maximal $c_i = \sum_{v \in T_i} a_v$ est choisie. L'heuristique *GreedyCostSeq* affecte les opérations à coût maximal à la même station.

Notons que l'heuristique gloutonne *GreedyCostSeq* peut aussi être utilisée pour résoudre approximativement le problème $P_{seq}(prec \mid k)$, tandis que l'heuristique randomisée *RandomNumberSeq* peut être utilisée pour résoudre approximativement le problème $P_{seq}(prec \mid k, cost)$.

Maintenant, la formulation de programmation linéaire en nombres entiers pour le cas général $P_{seq}(prec \mid k, cost)$ sera présentée. Supposons que le nombre minimal de stations k^* a été trouvé. Introduisons des variables 0-1 x_{ij} telles que $x_{ij} = 1$ si et seulement si une opération i est affectée à une station j.

Calculons la borne inférieure suivante $LBSeq_2$ sur le coût total minimal :

$$LBSeq_2 = \sum_{v=1}^{f} a_v \max \left\{ \left\lceil \frac{\sum_{i \in N_v} s_i}{r} \right\rceil, \left\lceil \frac{\sum_{i \in N_v} p_i}{U_v} \right\rceil \right\}.$$

Le problème $P_{seq}(prec \mid k, cost)$ peut être reformulé comme le problème de programmation

linéaire en nombres entiers suivant, noté comme $MinCostSeq$.

Problème *MinCostSeq*

$$\min \quad \sum_{j=1}^{k^*} \sum_{v=1}^{f} a_v z_{vj}, \tag{5.9}$$

sous les contraintes

$$\sum_{j=1}^{k^*} \sum_{v=1}^{f} a_v z_{vj} \geq LBSeq_2, \tag{5.10}$$

$$\sum_{j=1}^{k^*} x_{ij} = 1, \quad i = 1, \ldots, n, \tag{5.11}$$

$$\sum_{i=1}^{n} s_i x_{ij} \leq r, \quad j = 1, \ldots, k^*, \tag{5.12}$$

$$\sum_{i \in N_v} p_i x_{ij} \leq U_v z_{vj}, \ v = 1, \ldots, f, \ j = 1, \ldots, k^*, \tag{5.13}$$

$$\sum_{j=1}^{h} x_{bj} + \sum_{j=h+1}^{k^*} x_{aj} \leq 1, \ \forall (a,b) \in A, \ h = 1, \ldots, k^* - 1, \tag{5.14}$$

$$x_{ij}, z_{vj} \in \{0,1\}, \quad i = 1, \ldots, n, \ j = 1, \ldots, k^*, \ v = 1, \ldots, f. \tag{5.15}$$

Dans le problème $MinCostSeq$ il y a $k^*(f+n)$ variables et $1 + n + k^*(1 + f + |A|) - |A|$ contraintes sans les contraintes de type.

La contrainte (5.10) incorpore la borne inférieure $LBSeq_2$. Les contraintes (5.12) s'occupent de la capacité de stations. Les contraintes (5.13) garantissent que toute opération d'un certain type peut être affectée à une station si et seulement si cette station est ouverte pour ce type et si la borne supérieure sur le temps de cycle n'est pas dépassée. Les autres contraintes sont les mêmes que dans le problème $MinNumberSeq$.

5.7 Résultats numériques

Nous avons utilisé le solveur IBM ILOG Cplex version 12.2. pour résoudre les instances des problèmes $MinNumberSeq$ et $MinCostSeq$. Les expérimentations ont été effectuées sur un ordinateur avec un processeur Intel(R) Core(TM) Duo 2.2 GHz et une mémoire à accès direct de 4 Go sous le système d'exploitation MS Windows XP (32 bit).

Dans nos expérimentations, la limite du temps de résolution de chaque instance du $MinNumberSeq$ ou $MinCostSeq$ a été fixée à deux heures.

Trois familles d'instances ont été générées aléatoirement. Une famille est associée au nombre d'opérations $n \in \{40, 60, 80\}$. Pour toutes les instances, il y a trois types de produits. Les coûts de changements de séries sont $a_1 = 3$, $a_2 = 2$ et $a_3 = 1$. La capacité des stations est égale à $r = 10$.

Une taille $s_i \in \{1, 2, 3\}$, un temps opératoire $p_i \in \{s_i, 3s_i\}$ et un ensemble de types $T_i \in \left\{\{1,2,3\},\{1,2\},\{1,3\},\{2,3\},\{1\},\{2\},\{3\}\right\}$ sont arbitrairement affectés à chaque opération $i \in N$. Les bornes supérieures U_v, $v \in F$ sur le temps de cycle sont calculées de la façon suivante. Calculons une borne inférieure sur le nombre de stations qui effectuent les opérations de type v : $LB^{(v)} = \left\lceil \frac{\sum_{i \in N_v} s_i}{r} \right\rceil$. Calculons le temps total moyen des opérations de type v par station au regard de la borne inférieure $LB^{(v)}$: $\bar{P}^{(v)} = (\sum_{i \in N_v} p_i)/LB^{(v)}$,$v \in F$. Choisissons de manière aléatoire la borne supérieure sur le temps de cycle U_v de l'intervalle $[\bar{P}^{(v)}, 1.5\bar{P}^{(v)}]$, $v \in F$.

Chaque famille est divisé en quatre *séries* n_1, n_2, n_3 et n_4 avec une densité de contraintes de précédence faible, moyenne, élevée et très élevée, respectivement. Pour chaque série n_e, $e \in \{1, 2, 3, 4\}$, et chaque paire (a, b) d'opérations telle que $a < b$ et a et b ont au moins un type commun, il est décidé avec la probabilité p_e que a précède b. Les probabilités correspondantes sont $p\_1 = 0,2$, $p\_2 = 0,4$, $p\_3 = 0,6$ et $p\_4 = 0,8$. Vu que chaque arc du graphe de précédence obtenu passe toujours d'un sommet avec un indice plus petit à un sommet avec un indice plus grand, ce graphe est acyclique.

Chaque série se compose de 20 instances. Leurs caractéristiques moyennes sont données dans le Tableau 5.7. $\%Prec$ représente la densité des contraintes de précédence, définies comme $\frac{|A|}{\max \#\text{ arcs}} \times 100\% = \frac{2|A|}{n^2-n} \times 100\%$.

La procédure suivante a été utilisée pour résoudre chaque instance du problème.

Procédure de résolution pour le problème $P_{seq}(prec \mid k, cost)$:

1. Appliquer les heuristiques $GreedyNumberSeq$ et $GreedyCostSeq$ une fois et l'heuristique $RandomNumberSeq$ 1000 fois. Déterminer le nombre minimal de stations, k^0, obtenu par ces heuristiques.
2. Appliquer Cplex pour résoudre le problème $MinNumberSeq$. Déterminer le nombre des stations minimal k^*.

TABLEAU 5.3 – Caractéristiques numériques moyennes

Séries n_e	40_1	40_2	40_3	40_4	60_1	60_2	60_3	60_4	80_1	80_2	80_3	80_4
$\%Prec$	15,96%	29,82%	45,80%	60,24%	15,13%	30,02%	44,81%	61,39%	15,63%	30,13%	45,19%	61,34%
$\sum_{i=1}^{n} s_i$	81,15	79,35	79,70	81,40	121,90	118,95	118,30	121,00	162,10	159,55	160,40	162,25
$\sum_{i=1}^{n} p_i$	164,60	155,05	157,05	158,40	240,70	237,65	238,25	238,00	324,15	317,45	320,50	328,70
U_1	23,35	22,50	21,20	23,35	24,00	24,15	24,05	24,65	24,30	24,95	23,80	24,65
U_2	23,00	23,75	24,70	23,25	24,10	25,10	24,20	23,20	24,05	24,75	23,45	26,10
U_3	23,55	21,85	22,20	23,70	24,05	24,20	24,70	24,70	25,10	24,35	24,30	24,75

3. Appliquer Cplex pour résoudre le problème $MinCostSeq$ avec k^* stations.
4. Sélectionner la meilleure solution du problème $P_{seq}(prec \mid k^*, cost)$ parmi les solutions obtenues aux étapes 1, 2 et 3.

Les résultats possibles de chaque lancement de Cplex pour une instance I sont les suivants :

- Une solution optimale avec la valeur $Opt(I)$ est trouvée.
- Une solution approchée avec la valeur $Appr(I)$, une erreur relative connue $\%Gap(I)$ et une borne inférieure $LB(I)$ sont obtenues.
- Aucune solution admissible n'a été trouvée lorsque la limite de temps est atteinte.

Une erreur relative $\%G(I)$ est définie comme $\frac{Appr(I)-LB(I)}{LB(I)} \times 100\%$. Dans nos expérimentations le troisième cas n'a jamais eu lieu.

Le Tableau 5.4 montre le nombre d'instances du problème $MinCostSeq$ résolues à l'optimalité (ligne « *Opt* ») et approximativement (ligne « *Appr* ») pour chaque série. Les erreurs relatives moyennes (ligne « $\%Err.\ moy.$ ») et les pires erreurs relatives (ligne « $\%Err.\ pire$ ») sont aussi indiquées. L'erreur relative moyenne $\%Err.\ moy.$ est définie comme $\frac{1}{|n\_e|} \sum_{I \in n\_e} \%Gap(I)$ et la pire erreur $\%Err.\ pire$ comme $\max_{I \in n\_e} \{\%G(I)\}$ pour chaque série n_e,$e \in \{1, 2, 3, 4\}$. Notons que toutes les instances du problème $MinNumberSeq$ ont été résolues à l'optimalité dans l'intervalle de temps donné.

Les résultats des heuristiques pour les problèmes $P_{seq}(prec \mid k)$ et $P_{seq}(prec \mid k, cost)$ sont présentés dans les Tableaux 5.5 et 5.6, respectivement. Les erreurs relatives moyennes (lignes $\%Err.moy.$) et les pires erreurs relatives (lignes $\%Err.pire$) y sont données.

Les résultats expérimentaux du Tableau 5.4 montrent que les instances de $MinCostSeq$ avec

TABLEAU 5.4 – $MinCostSeq$: qualité des solutions

Séries n_e	40_1	40_2	40_3	40_4	60_1	60_2	60_3	60_4	80_1	80_2	80_3	80_4
Opt	20	20	20	20	7	20	20	20	0	17	20	20
Appr	0	0	0	0	13	0	0	0	20	3	0	0
%Err. moy.	-	-	-	-	10,69	-	-	-	23,56	5,16	-	-
%Err. pire	-	-	-	-	27,50	-	-	-	38,46	35,94	-	-

TABLEAU 5.5 – $P_{seq}(prec \mid k)$: résultats des heuristiques

Séries n_e	40_1	40_2	40_3	40_4	60_1	60_2	60_3	60_4	80_1	80_2	80_3	80_4
GreedyNumberSeq %Err. moy.	0,63	1,88	3,14	0,56	0,38	2,02	1,63	1,22	0,87	1,92	1,80	2,08
GreedyNumberSeq %Err. pire	12,50	12,50	14,29	11,11	7,69	8,33	8,33	5,88	6,67	6,25	6,25	5,88
RandomNumberSeq %Err. moy.	0,00	0,00	0,00	0,00	0,00	0,00	0,42	0,00	0,00	0,63	0,88	0,29
RandomNumberSeq %Err. pire	0,00	0,00	0,00	0,00	0,00	0,00	8,33	0,00	0,00	6,25	6,25	5,88
GreedyCostSeq %Err. moy.	3,13	3,61	4,33	2,92	1,64	3,62	4,07	2,02	2,06	3,72	2,98	2,36
GreedyCostSeq %Err. pire	12,50	12,50	14,29	12,50	9,09	8,33	8,33	8,33	6,25	6,67	12,50	6,25

TABLEAU 5.6 – $P_{seq}(prec \mid k, cost)$: résultats des heuristiques

Séries n_e	40_1	40_2	40_3	40_4	60_1	60_2	60_3	60_4	80_1	80_2	80_3	80_4
RandomNumberSeq %Err. moy.	18,50	8,88	6,08	4,20	24,85	12,96	7,23	3,89	55,19	20,25	6,87	4,34
RandomNumberSeq %Err. pire	30,56	19,44	12,50	17,50	35,56	24,07	24,14	12,90	67,31	62,96	13,70	12,64
GreedyCostSeq %Err. moy.	32,38	21,87	10,97	12,81	51,42	24,30	17,20	8,53	65,90	27,43	13,51	9,34
GreedyCostSeq %Err. pire	50,00	36,11	18,75	33,33	76,74	42,59	31,03	18,18	90,38	81,36	23,29	21,84

une densité de contraintes de précédence élevée et très élevée peuvent être facilement résolues à l'optimalité. Pour les instances avec une densité faible et moyenne, des solutions approchées peuvent être calculées avec une erreur relative d'au plus $23,65\%$. Les pires erreurs relatives sont dans l'intervalle de $27,50\%$ jusqu'à $38,46\%$.

Les heuristiques obtiennent une solution très vite. L'heuristique $RandomNumberSeq$ s'est avérée la meilleure parmi les heuristiques proposées. Dans les Tableaux 5.5 et 5.6 on peut voir que $RandomNumberSeq$ surclasse $GreedyNumberSeq$ et $GreedyCostSeq$ pour $P_{seq}(prec \mid k)$ ainsi que pour $P_{seq}(prec \mid k, cost)$. $RandomNumberSeq$ calcule des solutions pour $P_{seq}(prec \mid k)$ avec une erreur relative moyenne d'au plus $0,88\%$. Pour la plupart des instances avec 40 et 60 opérations, $RandomNumberSeq$ obtient les solutions optimales. La performance de $GreedyNumberSeq$ ne peut pas rivaliser avec l'heuristique $RandomNumberSeq$, parce que

RandomNumberSeq est exécuté 1000 fois afin de diversifier la recherche et explorer différentes parties de l'espace de solutions. Cependant, *GreedyNumberSeq* calcule des solutions pour $P_{seq}(prec \mid k)$ avec une erreur relative moyenne maximale de $3,14\%$ et la pire erreur relative d'au plus $14,29\%$. *GreedyCostSeq* manifeste la pire performance pour $P_{seq}(prec \mid k)$ par rapport aux autres heuristiques proposées. Cela est dû au fait que *GreedyCostSeq* a été initialement conçu pour résoudre approximativement des instances de $P_{seq}(prec \mid k, cost)$ et non celles de $P_{seq}(prec \mid k)$.

Pour le problème $P_{seq}(prec \mid k, cost)$, les erreurs relatives moyennes maximales, obtenues par *RandomNumberSeq* et *GreedyCostSeq*, sont égales à $55,19\%$ et $65,90\%$, respectivement. Selon nos observations, la qualité des solutions s'améliore avec l'augmentation de la densité des contraintes de précédence. Cette augmentation suscite la réduction du nombre de solutions admissibles. Ainsi, la performance des heuristiques est positivement corrélée avec le nombre de solutions admissibles.

Pour chaque série, les Tableaux 5.7 et 5.8 montrent le nombre d'instances des problèmes *MinNumberSeq* et *MinCostSeq*, respectivement, qui ont été résolues à l'optimalité dans l'intervalle de temps (en minutes et en heures) défini par la ligne. Notons que la résolution d'une instance des problèmes $P_{seq}(prec \mid k)$ et $P_{seq}(prec \mid k, cost)$ par une des heuristiques *GreedyNumberSeq*/*GreedyCostSeq* et *RandomNumberSeq* exige seulement 0,1 et 0,5 secondes en moyenne, respectivement.

On peut observer que le temps de calcul pour *MinNumberSeq* est positivement corrélé avec le nombre d'opérations et la densité des contraintes de précédence. Étant donné que nous calculons k^0 par une des heuristiques et que *RandomNumberSeq* fournit le nombre optimal de stations pour presque toutes les instances avec 40 jusqu'à 60 opérations, k^0 est égal à la valeur de la solution optimale pour toutes les instances respectives. Pour Cplex, il ne reste qu'à vérifier l'optimalité de la solution. Lorsque le nombre d'opérations augmente jusqu'à 80, la qualité de solution des heuristiques se réduit de sorte que le nombre optimal de stations n'est pas toujours obtenu. Cela augmente le temps de calcul de Cplex pour *MinNumberSeq*. En outre, lorsque la densité des contraintes de précédence augmente, l'affectation précise des opérations aux stations joue un rôle de plus en plus important dans le processus de résolution de

MinNumberSeq. Cette propriété, très probablement, augmente également le temps de calcul.

Pourtant, le temps de calcul pour *MinCostSeq* est négativement corrélé avec la densité des contraintes de précédence. Une faiblesse de la formulation de MinCost *MinCostSeq* consiste en la symétrie dans les variables de stations qui suscite une mauvaise performance de Cplex. Lorsque la densité des contrainte de précédence augmente, la symétrie des variable se réduit et donc la performance de Cplex s'améliore.

TABLEAU 5.7 – *MinNumberSeq* : le nombre des instances résolues à l'optimalité dans les intérvalles de temps

Séries n_e	40_1	40_2	40_3	40_4	60_1	60_2	60_3	60_4	80_1	80_2	80_3	80_4
$<$ 1min	20	20	20	20	20	20	20	20	20	20	7	5
[1min; 5min]	0	0	0	0	0	0	0	0	0	0	12	11
]5min; 30min]	0	0	0	0	0	0	0	0	0	0	1	3
$>$ 30min	0	0	0	0	0	0	0	0	0	0	0	1

TABLEAU 5.8 – *MinCostSeq* : le nombre des instances résolues à l'optimalité dans les intérvalles de temps

Séries n_e	40_1	40_2	40_3	40_4	60_1	60_2	60_3	60_4	80_1	80_2	80_3	80_4
$<$ 1min	20	20	20	20	4	20	20	20	0	11	19	19
[1min; 5min]	0	0	0	0	0	0		0	0	2	1	1
]5min; 30min]	0	0	0	0	1	0	0	0	0	3	0	0
$>$ 30min	0	0	0	0	2	0	0	0	0	1	0	0

Nous pouvons conclure que les approches de résolution proposées résolvent des instances du problème $P_{seq}(prec \mid k, cost)$ de dimension pratique en un temps acceptable. Comme dans le chapitre 4 nous rappelons qu'en considérant une taille moyenne des opérations qui est égale à deux, les instances les plus grandes contiennent donc 160 opérations unitaires.

5.8 Conclusion

Dans ce chapitre nous avons étudié un problème de configuration d'une ligne de fabrication multi-produit cadencée avec un mode d'exécution séquentielle des opérations sur les stations.

Le cheminement de production de cette ligne est unique pour tous les types de produits, et son schéma logique est linéaire. La fabrication de chaque type de produits exige un ensemble d'opérations spécifiques. Les produits de tous les types sont transportés entre les stations dans la même direction. Toutes les opérations de tous les types doivent être affectées aux stations. Un changement de séries s'effectue sur une station si au moins une opération d'un certain type est affectée à cette station. Les coûts et les temps de changements de séries dépendent du type de produits. Chaque opération implique l'emploi d'un ou de plusieurs outils standard. Le nombre des outils nécessaires pour effectuer une opération est appelé une « taille d'opération ». La taille totale des opérations sur une station est limitée. Des contraintes de précédence sont données sur l'ensemble des opérations. Les produits de tous les types sont ordonnés selon le principe de la *technologie de groupe*. Le temps de cycle est égal au temps de station maximal. Il y a une borne supérieure sur le temps de cycle pour chaque type de produit. Le problème a deux critères considérés lexicographiquement : la minimisation du nombre de stations et la minimisation du coût total de changements de séries.

La complexité de calcul pour plusieurs cas spéciaux de ce problème est donnée dans le Tableau 5.9.

Tableau 5.9 – Complexité de calcul pour les cas spéciaux du problème $P_{seq}(prec, excl, s_i \mid k, cost)$

Problème	Complexité
$P_{seq}(r = \infty \mid k)$	NP-difficile au sens fort
$P_{seq}(U = \infty \mid k)$	NP-difficile au sens fort
$P_{seq}(f = 3, p_i = 1, s_i = 1, chains, r = \infty \mid k)$	NP-difficile au sens fort
$P_{seq}(p_i = 1, s_i = 1, prec, U_v = \infty \mid k)$	$O(n + \|A\|)$
$P_{seq}(p_i = 1, s_i = 1, U_v = \infty \mid k)$	$O(1)$
$P_{seq}(p_i = 1, s_i = 1, U_v = \infty \mid k, cost)$	NP-difficile au sens fort, O(fonction de f)

Afin de résoudre le problème étudié, nous avons utilisé des heuristiques constructives et des formulations de programmation linéaire en nombres entiers. Les expérimentations numériques ont montré que les approches de résolution proposées sont capables de résoudre des instances

du problème de dimension pratique en un temps acceptable.

Conclusion générale

Cette thèse est consacrée à la problématique de l'optimisation combinatoire pour la configuration des lignes de fabrication. La configuration est une étape du processus de conception préliminaire des lignes de fabrication qui consiste à choisir les ressources et les affecter à des stations. Il est donc très important de configurer une ligne de fabrication de la meilleure façon possible car une erreur à ce stade coûte très cher. Ainsi, il est nécessaire de développer des approches d'optimisation combinatoire efficaces dédiées à ce sujet.

Dans le chapitre 1 nous avons distingué les lignes de fabrication des autres systèmes de production. Nous avons classé les lignes de fabrication suivant les critères de variabilité et de type de flux de produits. Nous avons analysé le processus de conception préliminaire des lignes de fabrication et défini le positionnement de la configuration de lignes dans ce processus.

Un état de l'art sur les problèmes de configuration de lignes de fabrication a été présenté dans le chapitre 2. Afin de présenter structurellement les hypothèses des problèmes examinés dans ce mémoire, nous avons introduit une nouvelle classification de problèmes de configuration de lignes. La spécificité de notre classification par rapport aux autres classifications existantes consiste à mettre un accent particulier sur la variabilité de production de lignes et leurs caractéristiques de temps et de coût. Une revue de la littérature a été présentée selon cette classification.

Le choix de notre recherche a été plus particulièrement motivé par les problèmes observés dans le domaine de l'usinage. L'un des problèmes, le problème d'équilibrage et de choix d'équipement pour des lignes dédiés (chapitre 3) est relativement peu étudié. Cela a présenté une opportunité de créer une nouvelle approche, plus performante que les approches antérieures. L'autre problème, concernant la minimisation des coûts de changements de séries pour des lignes multi-produits flexibles ou reconfigurables (chapitres 4 et 5), s'est avéré ne pas être étudié dans

la littérature.

Le problème examiné dans le chapitre 3 concerne non seulement l'équilibrage de ligne mais aussi la sélection des ressources à partir d'un ensemble donné de ressources. Afin de le résoudre, nous avons employé une approche qui se base sur la réduction de ce problème à un problème de partition d'ensemble à l'aide du concept de station localement admissible. Nous avons utilisé une procédure de prétraitement des données qui avait pour but de réduire le nombre de variables et l'espace des solutions admissibles. Des heuristiques gloutonnes et un algorithme de génération de contraintes ont été mis en œuvre. Les expérimentations numériques menées sur des données de référence ont prouvé l'efficacité de notre approche qui surclasse les méthodes de résolution connues dans la littérature à la fois en termes de qualité des solutions obtenues et d'effort de calcul.

Les chapitres suivants ont été consacrés à la minimisation des coûts de changements de séries. Les problèmes liés à la configuration optimale des lignes de fabrication minimisant le coût total de changements de séries n'ont pas été étudiés dans la littérature. Pourtant, dans l'industrie, la prise en compte des coûts de changements de séries est importante. Un changement de séries signifie l'ensemble des démarches nécessaires pour préparer une station au traitement d'un produit brut d'un certain type. Ces démarches incluent : le réglage des machines, le chargement d'une pièce de produit, le positionnement du produit et des ressources/outils, le déchargement et le nettoyage de la station. Le changement de séries engendre des coûts additionnels liés à la consommation d'énergie et à la présence d'opérateur.

Dans les chapitres 4 et 5 nous avons étudié deux problèmes de minimisation du coût total de changement de séries. Ils se distinguent notamment par le mode d'exécution des opérations sur les stations : l'un est parallèle et l'autre est séquentiel. La complexité de calcul pour certains cas de ces problèmes a été établie. La procédure générale proposée pour la résolution des problèmes est identique. Elle combine des heuristiques gloutonnes et un programme linéaire en nombre entiers utilisé avec le solveur Cplex. Les expérimentations numériques ont montré que notre approche est capable de résoudre des instances de dimension réelle en un temps acceptable.

Quant aux perspectives de notre recherche, nous nous intéressons particulièrement à l'étude de la stabilité des solutions obtenues. La nécessité de cette recherche vient du fait que les

données des problèmes sont souvent soumises à des perturbations qui apparaissent après la prise de la décision concernant la configuration de la ligne. Dans ce cas, une partie des paramètres deviennent incertains, et la qualité ou même la faisabilité des solutions n'est plus assurée. La prise en compte de ces conditions nécessite donc le développement des méthodes robustes. Les autres pistes de recherche peuvent consister en l'amélioration des méthodes proposées afin de résoudre des problèmes de taille supérieure ou en une généralisation des formulations proposées pour tenir compte d'autres contraintes ou d'autres spécifications existantes dans l'industrie.

Bibliographie

[gam, 2012] (2012). www.gams.com.

[w12, 2012] (2012). www.w124.org.

[Aĝpak and Gökçen, 2005] Aĝpak, K. and Gökçen, H. (2005). Assembly line balancing : two resource constrained cases. *International Journal of Production Economics*, 96(1) :129–140.

[Albareda-Sambola et al., 2011] Albareda-Sambola, M., Fernández, E., and Laporte, G. (2011). A computational comparison of several models for the exact solution of the capacity and distance constrained plant location problem. *Computers & Operations Research*, 38(8) :1109–1116.

[Allahverdi et al., 2008] Allahverdi, A., Ng, C. T., Cheng, T. C. E., and Kovalyov, M. Y. (2008). A survey of scheduling problems with setup times or costs. *European Journal of Operational Research*, 187(3) :985–1032.

[Amen, 2000] Amen, M. (2000). An exact method for cost-oriented assembly line balancing. *International Journal of Production Economics*, 64(1) :187–195.

[Amen, 2001] Amen, M. (2001). Heuristic methods for cost-oriented assembly line balancing : A comparison on solution quality and computing time. *International Journal of Production Economics*, 69(3) :255–264.

[Andrès et al., 2008] Andrès, C., Miralles, C., and Pastor, R. (2008). Balancing and scheduling tasks in assembly lines with sequence-dependent setup times. *European Journal of Operational Research*, 187(3) :1212–1223.

[Atmani, 1995] Atmani, A. (1995). A production planning model for flexible manufacturing systems with setup cost consideration. *Computers & Industrial Engineering*, 29(1-4) :723–727.

[Baldacci et al., 2011] Baldacci, R., Mingozzi, A., and Roberti, R. (2011). New route relaxation and pricing strategies for the vehicle routing problem. *Operations Research*, 59(5) :1269–1283.

[Bard, 1989] Bard, J. F. (1989). Assembly line balancing with parallel workstations and dead time. *International Journal of Production Research*, 27(6) :1005–1018.

[Battaïa and Dolgui, 2012] Battaïa, O. and Dolgui, A. (2012). Reduction approaches for a generalized line balancing problem. *Computers & Operations Research*, 39(10) :2337–2345.

[Battaïa et al., 2012] Battaïa, O., Dolgui, A., Guschinsky, N., and Levin, G. (2012). A decision support system for design of mass production machining lines composed of stations with rotary or mobile table. *Robotics and Computer Integrated Manufacturing*, 28(6) :672–680.

[Baybars, 1986] Baybars, I. (1986). A survey of exact algorithms for the simple assembly line balancing. *Management Science*, 32(8) :909–932.

[Becker and Scholl, 2006] Becker, C. and Scholl, A. (2006). A survey on problems and methods in generalized assembly line balancing. *European Journal of Operational Research*, 168(3) :694–715.

[Belmokhtar, 2006] Belmokhtar, S. (2006). *Lignes d'usinage avec équipements standard : modélisation, configuration et optimisation.* PhD thesis, École Nationale Supérieure des Mines de Saint-Étienne.

[Belmokhtar et al., 2006] Belmokhtar, S., Dolgui, A., Guschinsky, N., and Levin, G. (2006). Integer programming models for logical layout design of modular machining lines. *Computers & Industrial Engineering*, 51(3) :502–518.

[Belmokhtar et al., 2007] Belmokhtar, S., Dolgui, A., Ignatenko, I., and Delorme, X. (2007). Optimizing modular machining line design problem with mixed activation mode of machining units. *Decision Making in Manufacturing and Services*, 1(1-2) :35–48.

[Blum, 2008] Blum, C. (2008). Beam-ACO for simple assembly line balancing. *INFORMS Journal on Computing*, 20(4) :618–627.

[Boctor, 1995] Boctor, F. F. (1995). A multiple-rule heuristic for assembly line balancing. *Journal of the Operational Research Society*, 46(1) :62–69.

[Boysen and Fliedner, 2008] Boysen, N. and Fliedner, M. (2008). A versatile algorithm for assembly line balancing. *European Journal of Operational Research*, 184(1) :39–56.

[Boysen et al., 2007] Boysen, N., Fliedner, M., and Scholl, A. (2007). A classiffication of assembly line balancing problems. *European Journal of Operational Research*, 183(2) :674–693.

[Boysen et al., 2008] Boysen, N., Fliedner, M., and Scholl, A. (2008). Sequencing mixed-model assembly lines to minimize part inventory cost. *OR Spectrum*, 30(3) :611–633.

[Brucker and Knust, 2006] Brucker, P. and Knust, S. (2006). http ://www.informatik.uni-osnabrueck.de/knust/class/.

[Bukchin and Rubinovitz, 2003] Bukchin, J. and Rubinovitz, J. (2003). A weighted approach for assembly line design with station paralleling and equipment selection. *IIE Transactions*, 35(1) :73–85.

[Bukchin and Tzur, 2000] Bukchin, J. and Tzur, M. (2000). Design of flexible assembly line to minimize equipment cost. *IIE Transactions*, 32(7) :585–598.

[Burns and Daganzo, 1987] Burns, L. D. and Daganzo, C. F. (1987). Assembly line job sequencing principles. *International Journal of Production Research*, 25(1) :71–99.

[Buzacott and Shanthikumar, 1993] Buzacott, J. A. and Shanthikumar, J. G. (1993). *Stochastic Models of Manufacturing Systems*. Prentice Hall, Englewood Cliffs, NJ, 1st edition.

[Cakir et al., 2011] Cakir, B., Altiparmak, F., and Dengiz, B. (2011). Multi-objective optimization of a stochastic assembly line balancing : A hybrid simulated annealing algorithm. *Computers & Industrial Engineering*, 60(3) :376–384.

[Caprara and Pferschy, 2005] Caprara, A. and Pferschy, U. (2005). Modified subset sum heuristics for bin packing. *Information Processing Letters*, 96(1) :18–23.

[Chakravarty, 1988] Chakravarty, A. K. (1988). Line balancing with task learning effects. *IIE Transactions*, 20(2) :186–193.

[Chakravarty and Shtub, 1985] Chakravarty, A. K. and Shtub, A. (1985). Balancing mixed model lines with in-process inventories. *Management Science*, 31(9) :1161–1174.

[Chakravarty and Shtub, 1986] Chakravarty, A. K. and Shtub, A. (1986). A cost minimization procedure for mixed model production lines with normally distributed task times. *European Journal of Operational Research*, 23(1) :25–36.

[Chiang, 1998] Chiang, W.-C. (1998). The application of a tabu search metaheuristic to the assembly line balancing problem. *Annals of Operations Research*, 77(0) :209–227.

[Chica et al., 2011] Chica, M., Cordón, O., Damas, S., and Bautista, J. (2011). Including different kinds of preferences in a multi-objective ant algorithm for time and space assembly line balancing on different Nissan scenarios. *Expert Systems with Applications*, 38(1) :709–720.

[Corominas et al., 2008] Corominas, A., Pastor, R., and Plans, J. (2008). Balancing assembly line with skilled and unskilled workers. *Omega*, 36(6) :1126–1132.

[Daganzo and Blumenfeld, 1994] Daganzo, C. F. and Blumenfeld, D. E. (1994). Assembly system design principles and tradeoffs. *International Journal of Production Research*, 32(3) :669–681.

[Davida and Linton, 1976] Davida, G. I. and Linton, D. J. (1976). A new algorithm for the scheduling of tree structured tasks. In *Proc. Conf. Inform. Sci. and Syst. Baltimore, MD*, pages 543–548.

[Delorme et al., 2009] Delorme, X., Dolgui, A., Essafi, M., Linxe, L., and Poyard, D. (2009). *Springer Handbook of Automation*, chapter Machining Lines Automation, pages 599–618. Springer.

[Delorme et al., 2012] Delorme, X., Dolgui, A., and Kovalyov, M. Y. (2012). Combinatorial design of a minimum cost transfer line. *Omega*, 40(1) :31–41.

[Dimitriadis, 2006] Dimitriadis, S. G. (2006). Assembly line balancing and group working : A heuristic procedure for workers' groups operating on the same product and workstation. *Computers & Operations Research*, 33(9) :2757–2774.

[Dolgui et al., 2002] Dolgui, A., Eremeev, A., Kolokolov, A., and Sigaev, V. (2002). Buffer allocation in production line with unreliable machines. *Journal of Mathematical Modeling and Algorithms*, 1(2) :89–104.

[Dolgui et al., 2007] Dolgui, A., Eremeev, A., and Sigaev., V. (2007). HBBA : Hybrid algorithm for buffer allocation in tandem production lines. *Journal of Intelligent Manufacturing*, 18(3) :411–420.

[Dolgui et al., 2006a] Dolgui, A., Finel, B., Guschinsky, N., Levin, G., and Vernadat, F. (2006a). MIP approach to balancing transfer lines with blocks of parallel operations. *IIE Transactions*, 38(10) :869–882.

[Dolgui et al., 2005a] Dolgui, A., Finel, B., Vernadat, F., Guschinsky, N., and Levin, G. (2005a). A heuristic approach for transfer lines balancing. *Journal of Intelligent Manufacturing*, 16(2) :159–172.

[Dolgui et al., 1999] Dolgui, A., Guschinsky, N., and Levin, G. (1999). On problem of optimal design of transfer lines with parallel and sequential operations. In *Proceedings of the 7th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA'99)*, volume 1, pages 329–334.

[Dolgui et al., 2006b] Dolgui, A., Guschinsky, N., and Levin, G. (2006b). A special case of transfer lines balancing by graph approach. *European Journal of Operational Research*, 168(3) :732–746.

[Dolgui et al., 2005b] Dolgui, A., Guschinsky, N., Levin, G., Louly, M., and Belmokhtar, S. (2005b). Balancing of transfer lines with simultaneously activated spindles. In Kopacek, P., Morel, G., and Pereira, C., editors, *Information Control Problems In Manufacturing 2004 (2-Volume Set)*, pages 45–50. Elsevier Science.

[Dolgui et al., 2008] Dolgui, A., Guschinsky, N., Levin, G., and Proth, J.-M. (2008). Optimisation of multi-position machines and transfer lines. *European Journal of Operational Research*, 185(3) :1375–1389.

[Dolgui and Ihnatsenka, 2009a] Dolgui, A. and Ihnatsenka, I. (2009a). Balancing modular transfer lines with serial-parallel activation of spindle heads at stations. *Discrete Applied Mathematics*, 157(1) :68–89.

[Dolgui and Ihnatsenka, 2009b] Dolgui, A. and Ihnatsenka, I. (2009b). Branch and bound algorithm for a transfer line design problem : Stations with sequentially activated multi-spindle heads. *European Journal of Operational Research*, 197(3) :1119–1132.

[Dolgui and Proth, 2006] Dolgui, A. and Proth, J. M. (2006). *Les systèmes de production modernes. Volume 1 : Conception, gestion et optimisation.* Hermès Science.

[Duran Toksari et al., 2010] Duran Toksari, M., Işleyen, S. K., Güner, S. K., and Baykoç, O. F. (2010). Assembly line balancing problem with deterioration tasks and learning effect. *Expert Systems with Applications*, 37(2) :1223–1228.

[Easton, 1990] Easton, F. F. (1990). A dynamic program with fathoming and dynamic upper bounds for the assembly line balancing problem. *Computers & Operations Research*, 17(2) :163–175.

[Ege et al., 2009] Ege, Y., Azizoglu, M., and Ozdemirel, N. E. (2009). Assembly line balancing with station paralleling. *Computers & Industrial Engineering*, 57(4) :1218–1225.

[Erel and Sarin, 1998] Erel, E. and Sarin, S. C. (1998). A survey of the assembly line balancing procedures. *Production Planning & Control : The Management of Operations*, 9(5) :414–434.

[Essafi et al., 2012] Essafi, M., Delorme, X., and Dolgui, A. (2012). A reactive GRASP and path relinking for balancing reconfigurable transfer lines. *International Journal of Production Research*, 50(18) :5213–5238.

[Essafi et al., 2010] Essafi, M., Delorme, X., Dolgui, A., and Guschinskaya, O. (2010). A MIP approach for balancing transfer line with complex industrial constraints. *Computers & Industrial Engineering*, 58(3) :393–400.

[Falkenauer, 2005] Falkenauer, E. (2005). Line balancing in the real world. In *Proceedings of the International Conference on Product Lifecycle Management PLM'05*, pages 360–370.

[Fanjul-Peyro and Ruiz, 2012] Fanjul-Peyro, L. and Ruiz, R. (2012). Scheduling unrelated parallel machines with optional machines and jobs selection. *Computers & Operations Research*, 39(7) :1745–1753.

[Fernandez de la Vega and Lueker, 1981] Fernandez de la Vega, W. and Lueker, G. S. (1981). Bin packing can be solved within $(1 + \varepsilon)$ in linear time. *Combinatorica*, 1(4) :349–355.

[Gabow, 1982] Gabow, H. N. (1982). An almost linear algorithm for two processor scheduling. *Journal of the ACM*, 29(3) :766–780.

[Gadidov and Wilhelm, 2000] Gadidov, R. and Wilhelm, W. (2000). A cutting plane approach for the single-product assembly system design problem. *International Journal of Production Research*, 38(8) :1731–1754.

[Garey and Johnson, 1979] Garey, M. R. and Johnson, D. S. (1979). *Computers and Intractability : A Guide to the NP-Completeness*. W. H. Freeman & Company.

[Garey et al., 1976] Garey, M. R., Johnson, D. S., and Sethi, R. (1976). The complexity of flowshop and jobshop scheduling. *Mathematics of Operations Research*, 1(2) :117–129.

[Ghosh and Gagnon, 1989] Ghosh, S. and Gagnon, R. J. (1989). A comprehensive literature review and analysis of the design, balancing and scheduling of assembly systems. *International Journal of Production Research*, 27(4) :637–670.

[Gökçen et al., 2006] Gökçen, H., Ağpak, K., and Benzer, R. (2006). Balancing of parallel assembly lines. *International Journal of Production Economics*, 103(2) :600–609.

[Gunasekaran et al., 2001] Gunasekaran, A., McNeil, R., McGaughey, R., and Ajasa, T. (2001). Experiences of a small to medium size enterprise in the design and implementation of manufacturing cells. *International Journal of Computer Integrated Manufacturing*, 14(2) :212–223.

[Guschinskaya and Dolgui, 2009] Guschinskaya, O. and Dolgui, A. (2009). Comparison of exact and heuristic methods for a transfer line balancing problem. *International Journal of Production Economics*, 120(2) :276–286.

[Guschinskaya and Dolgui, 2010] Guschinskaya, O. and Dolgui, A. (2010). Équilibrage de lignes de production. *Journal Européen des Systèmes Automatisés*, 44(9-10) :1081–1119.

[Guschinskaya et al., 2009] Guschinskaya, O., Dolgui, A., Guschinsky, N., and Levin, G. (2009). Minimizing makespan for multi-spindle head machines with a mobile table. *Computers & Operations Research*, 36(2) :344–357.

[Hamta et al., 2011] Hamta, N., Fatemi Ghomi, S. M. T., Jolai, F., and Bahalke, U. (2011). Bicriteria assembly line balancing by considering flexible operation times. *Applied Mathematical Modelling*, 35(12) :5592–5608.

[Hamzadayi and Yildiz, 2012] Hamzadayi, A. and Yildiz, G. (2012). A genetic algorithm based approach for simultaneously balancing and sequencing of mixed-model U-lines with parallel workstations and zoning constraints. *Computers & Industrial Engineering*, 62(1) :206–215.

[Haouari and Serairi, 2009] Haouari, M. and Serairi, M. (2009). Heuristics for the variable sized bin-packing problem. *Computers & Operations Research*, 36(10) :2877–2884.

[Held et al., 1963] Held, M., Karp, R. M., and Shareshian, R. (1963). Assembly-line balancing - dynamic programming with precedence constraints. *Operations Research*, 11(3) :442–459.

[Hoffmann, 1992] Hoffmann, T. R. (1992). Eureka : A hybrid system for assembly line balancing. *Management Science*, 38(1) :39–47.

[Hu, 1961] Hu, T. C. (1961). Parallel sequencing and assembly line problems. *Operations Research*, 9(6) :841–848.

[Hwang et al., 2008] Hwang, R. K., Katayama, H., and Gen, M. (2008). U-shaped assembly line balancing problem with genetic algorithm. *International Journal of Production Research*, 46(16) :4637–4649.

[Jackson, 1956] Jackson, J. R. (1956). A computing procedure for a line balancing problem. *Management Science*, 2(3) :261–271.

[Johnson, 1988] Johnson, R. V. (1988). Optimally balancing large assembly lines with « FABLE ». *Management Science*, 34(2) :240–253.

[Kahn, 1962] Kahn, A. B. (1962). Topological sorting of large networks. *Communications of the ACM*, 5(11) :558–562.

[Karimi-Nasab et al., 2012] Karimi-Nasab, M., Bahalke, U., Feili, H. R., Sheikhzadeh, A., and Dolatkhahi, K. (2012). Working time evaluation in assembly lines. *International Journal of Mathematics in Operational Research*, 4(1) :1 – 17.

[Khan and Day, 2002] Khan, A. and Day, A. J. (2002). A knowledge based design methodology for manufacturing assembly lines. *Computers & Industrial Engineering*, 41(4) :441–467.

[Kilincci, 2011] Kilincci, O. (2011). Firing sequences backward algorithm for simple assembly line balancing problem of type 1. *Computers & Industrial Engineering*, 60(4) :830–839.

[Kim et al., 2009] Kim, Y. K., Song, W. S., and Kim, J. H. (2009). A mathematical model and a genetic algorithm for two-sided assembly line balancing. *Computers & Operations Research*, 36(3) :853–865.

[Kimms, 2000] Kimms, A. (2000). Minimal investment budgets for flow line configuration. *IIE Transactions*, 32(4) :287–298.

[Klein and Scholl, 1996] Klein, R. and Scholl, A. (1996). Maximizing the production rate in simple assembly line balancing - a branch and bound procedure. *European Journal of Operational Research*, 91(2) :367–385.

[Koren et al., 1999] Koren, Y., Heisel, U., Jovane, F., Moriwaki, T., Pritschow, G., Ulsoy, G., and Van Brussel, H. (1999). Reconfigurable manufacturing systems. *CIRP Annals*, 48(2) :527–540.

[Kottas and Lau, 1976] Kottas, J. F. and Lau, H. S. (1976). A total operating cost model for paced lines with stochastic task times. *AIIE Transactions*, 8(2) :234–240.

[Kovalev et al., 2012] Kovalev, S., Delorme, X., and Dolgui, A. (2012). Line configuration to minimize setup costs. *Mathematical and Computer Modelling*, 55(9-10) :2087–2095.

[Lapierre et al., 2006] Lapierre, S. D., Ruiz, A., and Soriano, P. (2006). Balancing assembly lines with tabu search. *European Journal of Operational Research*, 168(3) :826–837.

[Li et al., 2011] Li, S., Wang, H., Jack Hu, S., Lin, Y.-T., and Abell, J. A. (2011). Automatic generation of assembly system configuration with equipment selection for automotive battery manufacturing. *Journal of Manufacturing Systems*, 30(4) :188–195.

[Lucertini et al., 1998] Lucertini, M., Pacciarelli, D., and Pacifici, A. (1998). Modeling an assembly line for configuration and flow management. *Computer Integrated Manufacturing Systems*, 11(1-2) :15–24.

[Lutz et al., 1998] Lutz, C. M., Roscoe Davis, K., and Sunc, M. (1998). Determining buffer location and size in production lines using tabu search. *European Journal of Operational Research*, 106(2-3) :301–316.

[MacGregor Smith and Daskalaki, 1988] MacGregor Smith, J. and Daskalaki, S. (1988). Buffer space allocation in automated assembly lines. *Operations Research*, 36(2) :343–358.

[Malakooti, 1994] Malakooti, B. B. (1994). Assembly line balancing with buffers by multiple criteria optimization. *International Journal of Production Research*, 32(9) :2159–2178.

[Massim et al., 2010] Massim, Y., Yalaoui, F., Chatelet, L. A. E., and Zeblah, A. (2010). Efficient combined immune-decomposition algorithm for optimal buffer allocation in production lines for throughput and profit maximization. *Computers & Operations Research*, 37(4) :611–620.

[McMullen and Frazier, 1998] McMullen, P. R. and Frazier, G. V. (1998). Using simulated annealing to solve a multiobjective assembly line balancing problem with parallel workstations. *International Journal of Production Research*, 36(10) :2717–2741.

[Meyr, 2004] Meyr, H. (2004). Supply chain planning in the German automotive industry. *OR Spectrum*, 26(4) :447–470.

[Miltenburg and Wijngaard, 1994] Miltenburg, J. and Wijngaard, J. (1994). The U-line line balancing problem. *Management Science*, 40(10) :1378–1388.

[Miralles et al., 2007] Miralles, C., García-Sabater, J. P., Andrés, C., and Cardos, M. (2007). Advantages of assembly lines in sheltered work centres for disabled. A case study. *International Journal of Production Economics*, 110(1-2) :187–197.

[Moodie and Young, 1965] Moodie, C. L. and Young, H. H. (1965). A heuristic method of assembly line balancing for assumptions of constant or variable work element times. *Journal of Industrial Engineering*, 16(1) :23–29.

[Mozdgir et al., 2013] Mozdgir, A., Mahdavi, I., Seyedi Badeleh, I., and Solimanpur, M. (2013). Using the Taguchi method to optimize the differential evolution algorithm parameters for minimizing the workload smoothness index in simple assembly line balancing. *Mathematical and Computer Modelling*, 57(1-2) :137–151.

[Musharavati and Hamouda, 2012] Musharavati, F. and Hamouda, A. S. M. (2012). Enhanced simulated-annealing-based algorithms and their applications to process planning in reconfigurable manufacturing systems. *Advances in Engineering Software*, 45(1) :80–90.

[Nearchou, 2007] Nearchou, A. C. (2007). Balancing large assembly lines by a new heuristic based on differential evolution method. *The International Journal of Advanced Manufacturing Technology*, 34(9-10) :1016–1029.

[Nemhauser and Wolsey, 1988] Nemhauser, G. L. and Wolsey, L. A. (1988). *Integer and combinatorial optimization*. John Wiley and Sons, New York.

[Nicosia et al., 2002] Nicosia, G., Pacciarelli, D., and Pacifici, A. (2002). Optimally balancing assembly lines with different workstations. *Discrete Applied Mathematics*, 118(1-2) :99–113.

[Özcan, 2010] Özcan, U. (2010). Balancing stochastic two-sided assembly lines : A chance-constrained, piecewise-linear, mixed integer program and a simulated annealing algorithm. *European Journal of Operational Research*, 205(1) :81–97.

[Özcan and Toklu, 2009a] Özcan, U. and Toklu, B. (2009a). Balancing of mixed-model two-sided assembly lines. *Computers & Industrial Engineering*, 57(1) :217–227.

[Özcan and Toklu, 2009b] Özcan, U. and Toklu, B. (2009b). Multiple-criteria decision-making in two-sided assembly line balancing : A goal programming and a fuzzy goal programming models. *Computers & Operations Research*, 36(6) :1955–1965.

[Özcan and Toklu, 2009c] Özcan, U. and Toklu, B. (2009c). A tabu search algorithm for two-sided assembly line balancing. *The International Journal of Advanced Manufacturing Technology*, 43(7-8) :822–829.

[Pastor et al., 2002] Pastor, R., Andrés, C., Duran, A., and Pérez, M. (2002). Tabu search algorithms for an industrial multi-product and multi-objective assembly line balancing problem, with reduction of the task dispersion. *Journal of the Operational Research Society*, 53(12) :1317–1323.

[Pastor and Ferrer, 2009] Pastor, R. and Ferrer, L. (2009). An improved mathematical program to solve the simple assembly line balancing problem. *International Journal of Production Research*, 47(11) :2943–2959.

[Petrovic et al., 2005] Petrovic, S., Fayad, C., and Petrovic, D. (2005). Job shop scheduling with lot-sizing and batching in an uncertain real-world environment. In Kendall, G., Lei, L., and Pinedo, M., editors, *Proceedings of the 2nd Multidisciplinary International Conference on Scheduling : Theory and Applications (MISTA 2005)*, pages 363–379.

[Pinedo and Chao, 1999] Pinedo, M. and Chao, X. (1999). *Operations Scheduling with Applications in Manufacturing and Services*. Irwin/McGraw-Hill.

[Pinnoi and Wilhelm, 1998] Pinnoi, A. and Wilhelm, W. E. (1998). Assembly system design : A branch and cut approach. *Management Science*, 44(1) :103–118.

[Pinto et al., 1983] Pinto, P. A., Dannenbring, D. G., and Khumawala, B. M. (1983). Assembly line balancing with processing alternatives : An application. *Management Science*, 29(7) :817–830.

[Pollard et al., 2008] Pollard, D., Chuo, S., and Lee, B. (2008). Strategies for mass customization. *Journal of Business & Economics Research*, 6(7) :77–86.

[Rabbani et al., 2012] Rabbani, M., Kazemi, S. M., and Manavizadeh, N. (2012). Mixed model U-line balancing type-1 problem : A new approach. *Journal of Manufacturing Systems*, 31(2) :131–138.

[Radharamanan, 1994] Radharamanan, R. (1994). Group technology concepts as applied to flexible manufacturing systems. *International Journal of Production Economics*, 33(1-3) :133–142.

[Rekiek et al., 2002] Rekiek, B., Dolgui, A., Delchambre, A., and Bratcu, A. (2002). State of art of optimization methods for assembly line design. *Annual Reviews in Control*, 26(2) :163–174.

[Rosenberg and Ziegler, 1992] Rosenberg, O. and Ziegler, H. (1992). A comparison of heuristic algorithms for cost-oriented assembly line balancing. *Zeitschrift für Operations Research*, 36(6) :477–495.

[Rubinovitz and Levitin, 1995] Rubinovitz, J. and Levitin, G. (1995). Genetic algorithm for assembly line balancing. *International Journal of Production Economics*, 41(1) :343–354.

[Sabuncuoglu et al., 2009] Sabuncuoglu, I., Erel, E., and Alp, A. (2009). Ant colony optimization for the single model U-type assembly line balancing problem. *International Journal of Production Economics*, 120(2) :287–300.

[Salveson, 1955] Salveson, M. E. (1955). The assembly line balancing problem. *The Journal of Industrial Engineering*, 6(3) :18–25.

[Sarin et al., 1999] Sarin, S. C., Erel, E., and Dar-El, E. M. (1999). A methodology for solving single-model, stochastic assembly line balancing problem. *Omega*, 27(5) :525–535.

[Scholl, 1999] Scholl, A. (1999). *Balancing and sequencing of assembly lines*. Physica-Verlag Heidelberg, 2 edition.

[Scholl and Becker, 2005] Scholl, A. and Becker, C. (2005). A note on « an exact method for cost-oriented assembly line balancing ». *International Journal of Production Economics*, 97(3) :343–352.

[Scholl and Boysen, 2009] Scholl, A. and Boysen, N. (2009). Designing parallel assembly lines with split workplaces : Model and optimization procedure. *International Journal of Production Economics*, 119(1) :90–100.

[Scholl et al., 2008] Scholl, A., Boysen, N., and Fliedner, M. (2008). The sequence-dependent assembly line balancing problem. *OR Spectrum*, 30(3) :579–609.

[Scholl et al., 2010] Scholl, A., Fliedner, M., and Boysen, N. (2010). ABSALOM : Balancing assembly lines with assignment restrictions. *European Journal of Operational Research*, 200(3) :688–701.

[Scholl and Klein, 1997] Scholl, A. and Klein, R. (1997). SALOME : A bidirectional branch-and-bound procedure for assembly line balancing. *INFORMS Journal on Computing*, 9(4) :319–334.

[Scholl and Klein, 1999] Scholl, A. and Klein, R. (1999). Balancing assembly lines effectively - A computational comparison. *European Journal of Operational Research*, 114(1) :50–58.

[Shafer et al., 2001] Shafer, S. M., Nembhard, D. A., and Uzumeri, M. V. (2001). The effects of worker learning, forgetting, and heterogeneity on assembly line productivity. *Management Science*, 47(12) :1639–1653.

[Shi and Gershwin, 2009] Shi, C. and Gershwin, S. B. (2009). An efficient buffer design algorithm for production line profit maximization. *International Journal of Production Economics*, 122(2) :725–740.

[Simaria and Vilarinho, 2009] Simaria, A. S. and Vilarinho, P. M. (2009). 2-ANTBAL : An antcolony optimisation algorithm for balancing two-sided assembly lines. *Computers & Industrial Engineering*, 56(2) :489–506.

[Son et al., 2001] Son, S.-Y., Olsen, T. L., and Yip-Hoi, D. (2001). An approach to scalability and line balancing for reconfigurable manufacturing systems. *Integrated Manufacturing Systems*, 12(7) :500–511.

[Süer, 1998] Süer, G. A. (1998). Designing parallel assembly lines. *Computers & Industrial Engineering*, 35(3-4) :467–470.

[Talbot et al., 1986] Talbot, F. B., Patterson, J. H., and Gehrlein, W. V. (1986). A comparative evaluation of heuristic line balancing techniques. *Management Science*, 32(4) :430–454.

[Tanaev et al., 1994] Tanaev, V., Sotskov, Y., and Strusevich, V. (1994). *Scheduling Theory. Multi-Stage Systems*. Kluwer Academic Publishers.

[Tatikonda and Wemmerlöv, 1992] Tatikonda, M. V. and Wemmerlöv, U. (1992). Adoption and implementation of group technology classification and coding systems : insights from seven case studies. *International Journal of Production Research*, 30(9) :2087–2110.

[Thomopoulos and Lehman, 1969] Thomopoulos, N. T. and Lehman, M. (1969). The mixed model learning curve. *AIIE Transactions*, 1(2) :127–132.

[Tiacci, 2012] Tiacci, L. (2012). Event and object oriented simulation to fast evaluate operational objectives of mixed model assembly lines problems. *Simulation Modelling Practice and Theory*, 24 :35–48.

[Topaloglu et al., 2012] Topaloglu, S., Salum, L., and Ayca Supciller, A. (2012). Rule-based modeling and constraint programming based solution of the assembly line balancing problem. *Expert Systems with Applications*, 39(3) :3484–3493.

[Ullman, 1975] Ullman, J. D. (1975). NP-complete scheduling problems. *Journal of Computer and System Sciences*, 10(3) :384–393.

[Vilarinho and Simaria, 2002] Vilarinho, P. M. and Simaria, A. S. (2002). A two-stage heuristic method for balancing mixed-model assembly lines with parallel workstations. *International Journal of Production Research*, 40(6) :1405–1420.

[Wang and Koren, 2012] Wang, W. and Koren, Y. (2012). Scalability planning for reconfigurable manufacturing systems. *Journal of Manufacturing Systems*, 31(2) :83–91.

[Wee and Magazine, 1982] Wee, T. S. and Magazine, M. J. (1982). Assembly line balancing as generalized bin packing. *Operations Research Letters*, 1(2) :56–58.

[Wei and Chao, 2011] Wei, N.-C. and Chao, I.-M. (2011). A solution procedure for type E simple assembly line balancing problem. *Computers & Industrial Engineering*, 61(3) :824–830.

[Wild, 1998] Wild, R. (1998). *Production and Operations Management*. Cassel, 5th edition.

[Xu and Xiao, 2011] Xu, W. and Xiao, T. (2011). Strategic robust mixed model assembly line balancing based on scenario planning. *Tsinghua Science and Technology*, 16(3) :308–314.

[Yagmahan, 2011] Yagmahan, B. (2011). Mixed-model assembly line balancing using a multi-objective ant colony optimization approach. *Expert Systems with Applications*, 38(10) :12453–12461.

[Yokoyama, 2008] Yokoyama, M. (2008). Flow-shop scheduling with setup and assembly operations. *European Journal of Operational Research*, 187(3) :1184–1195.

[Yoosefelahi et al., 2012] Yoosefelahi, A., Aminnayeri, M., Mosadegh, H., and Davari Ardakani, H. (2012). Type II robotic assembly line balancing problem : An evolution strategies algorithm for a multi-objective model. *Journal of Manufacturing Systems*, 31(2) :139–151.

École Nationale Supérieure des Mines
de Saint-Étienne

NNT : 2012 EMSE 0672

Sergey KOVALEV

COMBINATORIAL PROBLEMS IN PRODUCTION LINES CONFIGURATION: COMPUTATIONAL ANALYSIS AND OPTIMIZATION

Speciality : Industrial Engineering

Keywords : combinatorial optimization, production lines, assembly line balancing, computational complexity

Abstract :

The objective of this thesis is to create and develop new effective solution methods for production line configuration problems. Two problems were studied: the equipment selection and balancing problem for dedicated lines and the setup cost minimization problem for multi-product lines. A solution for the first problem consists in a feasible assignment of the resources to an unknown number of stations so that the total cost is minimized. In order to solve this problem, we reduced it to the set partitioning problem and solved it by greedy heuristics and an exact method of constraint generation. The computer experiments on different problem instances showed that the new solution approach outperforms the previous methods from the literature both in terms of solution quality and computational time. For the second problem two criteria were considered lexicographically: the minimization of the number of stations and the minimization of the total setup cost. We examined successively the cases with parallel and sequential execution of operations. Approximate solutions were found by greedy heuristics. Then, we proposed two integer programming models in order to obtain the minimal number of stations and then the minimal setup cost. The experimental results for this new problem proved to be promising both in terms of solution quality and computational time.

École Nationale Supérieure des Mines
de Saint-Étienne

NNT : 2012 EMSE 0672

Sergey KOVALEV

PROBLÈMES COMBINATOIRES EN CONFIGURATION DES LIGNES DE FABRICATION : ANALYSE DE COMPLEXITÉ ET OPTIMISATION

Spécialité: Génie industriel

Mots clefs : optimisation combinatoire, configuration de lignes de fabrication, équilibrage de lignes, complexité de calcul

Résumé :

L'objectif de la thèse est de créer et développer de nouvelles méthodes de résolution efficaces des problèmes combinatoires en configuration des lignes de fabrication. Deux problèmes ont été particulièrement étudiés: le problème d'équilibrage et de choix d'équipement pour des lignes dédiées et le problème de minimisation des coûts de changements de séries pour des lignes multi-produits. Une solution du premier problème consiste en une affectation admissible des ressources à un nombre de stations à déterminer de sorte que le coût total soit minimal. Afin de résoudre ce problème, nous l'avons réduit au problème de partition d'ensemble et l'avons résolu par des heuristiques gloutonnes et une méthode exacte de génération de contraintes. Les expérimentations sur différentes instances ont montré que la nouvelle approche de résolution surclasse les approches antérieures de la littérature en termes de qualité de solution et de temps de calcul. Pour le second problème deux critères sont considérés lexicographiquement : la minimisation du nombre de stations et la minimisation du coût de changement de séries. Nous avons examiné successivement les cas d'exécution parallèle et séquentielle des opérations. Des solutions approchées ont été trouvées par des heuristiques gloutonnes. Ensuite, nous avons proposé deux modèles de programmation linéaire en nombres entiers (PLNE) afin de trouver le nombre de stations minimal et ensuite d'obtenir le coût de changement de séries minimal. Les résultats des expérimentations sur ces nouveaux problèmes se sont avérés prometteurs à la fois en termes de qualité de solution et de temps de calcul.