Conception et optimisation d'allocation de ressources dans les lignes d'usinage reconfigurables

Mohamed Essafi

NNT : 2010 EMSE 0588

THÈSE

présentée par

Mohamed ESSAFI

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

Spécialité : Génie Industriel

Conception et optimisation d'allocation de ressources dans les lignes d'usinage reconfigurables

Soutenance prévue le 08 Décembre 2010 à Saint-Étienne

Membres du jury

Rapporteurs :	Jean-Luc PARIS	Professeur, IFMA, Clermont Ferrand
	Farouk YALAOUI	Professeur, UTT, Troyes
Examinateurs :	Abdelaziz BOURAS	Professeur, Université Lumière Lyon 2, Lyon
	Albert COROMINAS	Professeur, UPC, Barcelone
	Anatol PASHKEVICH	Professeur, EMN, Nantes
	Serge TICHKIEWITCH	Professeur, G-SCOP, INP Grenoble
Directeurs de thèse :	Alexandre DOLGUI	Professeur, ENSMSE, Saint-Étienne
	Xavier DELORME	Maître-Assistant, ENSMSE, Saint-Étienne
Invité :	Damien POYARD	Directeur de PCI-SCEMM, Saint-Étienne

Spécialités doctorales :	**Responsables :**
SCIENCES ET GENIE DES MATERIAUX	J. DRIVER Directeur de recherche – Centre SMS
MECANIQUE ET INGENIERIE	A. VAUTRIN Professeur – Centre SMS
GENIE DES PROCEDES	G. THOMAS Professeur – Centre SPIN
SCIENCES DE LA TERRE	B. GUY Maître de recherche – Centre SPIN
SCIENCES ET GENIE DE L'ENVIRONNEMENT	J. BOURGOIS Professeur – Centre SITE
MATHEMATIQUES APPLIQUEES	E. TOUBOUL Ingénieur – Centre G2I
INFORMATIQUE	O. BOISSIER Professeur – Centre G2I
IMAGE, VISION, SIGNAL	JC. PINOLI Professeur – Centre CIS
GENIE INDUSTRIEL	P. BURLAT Professeur – Centre G2I
MICROELECTRONIQUE	Ph. COLLOT Professeur – Centre CMP

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

AVRIL	Stéphane	MA	Mécanique & Ingénierie	CIS
BATTON-HUBERT	Mireille	MA	Sciences & Génie de l'Environnement	SITE
BENABEN	Patrick	PR 1	Sciences & Génie des Matériaux	CMP
BERNACHE-ASSOLLANT	Didier	PR 0	Génie des Procédés	CIS
BIGOT	Jean-Pierre	MR	Génie des Procédés	SPIN
BILAL	Essaïd	DR	Sciences de la Terre	SPIN
BOISSIER	Olivier	PR 1	Informatique	G2I
BORBELY	Andras	MR	Sciences et Génie des Matériaux	SMS
BOUCHER	Xavier	MA	Génie Industriel	G2I
BOUDAREL	Marie-Reine	PR 2	Génie Industriel	DF
BOURGOIS	Jacques	PR 0	Sciences & Génie de l'Environnement	SITE
BRODHAG	Christian	DR	Sciences & Génie de l'Environnement	SITE
BURLAT	Patrick	PR 2	Génie industriel	G2I
COLLOT	Philippe	PR 1	Microélectronique	CMP
COURNIL	Michel	PR 0	Génie des Procédés	SPIN
DAUZERE-PERES	Stéphane	PR 1	Génie industriel	CMP
DARRIEULAT	Michel	IGM	Sciences & Génie des Matériaux	SMS
DECHOMETS	Roland	PR 1	Sciences & Génie de l'Environnement	SITE
DESRAYAUD	Christophe	MA	Mécanique & Ingénierie	SMS
DELAFOSSE	David	PR 1	Sciences & Génie des Matériaux	SMS
DOLGUI	Alexandre	PR 1	Génie Industriel	G2I
DRAPIER	Sylvain	PR 2	Mécanique & Ingénierie	SMS
DRIVER	Julian	DR 0	Sciences & Génie des Matériaux	SMS
FEILLET	Dominique	PR 2	Génie Industriel	CMP
FOREST	Bernard	PR 1	Sciences & Génie des Matériaux	CIS
FORMISYN	Pascal	PR 1	Sciences & Génie de l'Environnement	SITE
FORTUNIER	Roland	PR 1	Sciences & Génie des Matériaux	SMS
FRACZKIEWICZ	Anna	DR	Sciences & Génie des Matériaux	SMS
GARCIA	Daniel	MR	Génie des Procédés	SPIN
GIRARDOT	Jean-Jacques	MR	Informatique	G2I
GOEURIOT	Dominique	MR	Sciences & Génie des Matériaux	SMS
GRAILLOT	Didier	DR	Sciences & Génie de l'Environnement	SITE
GROSSEAU	Philippe	MR	Génie des Procédés	SPIN
GRUY	Frédéric	MR	Génie des Procédés	SPIN
GUY	Bernard	MR	Sciences de la Terre	SPIN
GUYONNET	René	DR	Génie des Procédés	SPIN
HERRI	Jean-Michel	PR 2	Génie des Procédés	SPIN
INAL	Karim	PR 2	Microélectronique	CMP
KLÖCKER	Helmut	DR	Sciences & Génie des Matériaux	SMS
LAFOREST	Valérie	CR	Sciences & Génie de l'Environnement	SITE
LERICHE	Rodolphe	CR CNRS	Mécanique et Ingénierie	SMS
LI	Jean-Michel	EC (CCI MP)	Microélectronique	CMP
LONDICHE	Henry	MR	Sciences & Génie de l'Environnement	SITE
MALLIARAS	George Grégory	PR 1	Microélectronique	CMP
MOLIMARD	Jérôme	MA	Mécanique et Ingénierie	SMS
MONTHEILLET	Frank	DR 1 CNRS	Sciences & Génie des Matériaux	SMS
PERIER-CAMBY	Laurent	PR 2	Génie des Procédés	SPIN
PIJOLAT	Christophe	PR 1	Génie des Procédés	SPIN
PIJOLAT	Michèle	PR 1	Génie des Procédés	SPIN
PINOLI	Jean-Charles	PR 0	Image, Vision, Signal	CIS
STOLARZ	Jacques	CR	Sciences & Génie des Matériaux	SMS
SZAFNICKI	Konrad	MR	Sciences & Génie de l'Environnement	SITE
THOMAS	Gérard	PR 0	Génie des Procédés	SPIN
TRIA	Assia		Microélectronique	CMP
VALDIVIESO	François	MA	Sciences & Génie des Matériaux	SMS
VAUTRIN	Alain	PR 0	Mécanique & Ingénierie	SMS
VIRICELLE	Jean-Paul	MR	Génie des procédés	SPIN
WOLSKI	Krzysztof	DR	Sciences & Génie des Matériaux	SMS
XIE	Xiaolan	PR 1	Génie industriel	CIS

Glossaire :

PR 0	Professeur classe exceptionnelle
PR 1	Professeur 1ère classe
PR 2	Professeur 2ème classe
MA(MDC)	Maître assistant
DR	Directeur de recherche
Ing.	Ingénieur
MR(DR2)	Maître de recherche
CR	Chargé de recherche
EC	Enseignant-chercheur
IGM	Ingénieur général des mines

Centres :

SMS	Sciences des Matériaux et des Structures
SPIN	Sciences des Processus Industriels et Naturels
SITE	Sciences Information et Technologies pour l'Environnement
G2I	Génie Industriel et Informatique
CMP	Centre de Microélectronique de Provence
CIS	Centre Ingénierie et Santé

Dernière mise à jour le : 13 septembre 2010

Lorsqu'un jour, le peuple aspire à vivre

Le destin se doit de répondre !

Les ténèbres se dissiperont !

Et les chaînes se briseront !

Abou el Kacem Chebbi

Remerciements

Au terme de ce travail, je tiens tout particulièrement à remercier Monsieur Alexandre Dolgui et Monsieur Xavier Delorme, pour leur assistance et leur disponibilité. Grâce à eux, ces trois années qu'on constitué ma thèse, ont été pleine d'enseignements, de beaux souvenirs et de travail.

J'exprime ma profonde gratitude aux membres du jury, qui ont accepté d'évaluer mon travail de thèse. Je remercie mes deux rapporteurs, Monsieur Jean-Luc PARIS et Monsieur Farouk YALAOUI pour leurs remarques et leurs rapports qui me sont d'une grande valeur. Merci à Monsieur Serge TICHKIEWITCH d'avoir accepté de présider le jury de cette thèse. Merci également à Monsieur Abdelaziz BOURAS, Monsieur Albert COROMINAS et Monsieur Anatol PASHKEVICH, pour avoir accepté d'examiner mon mémoire et de faire partie de mon jury de thèse. Je tiens également à remercier Monsieur Damien POYARD, directeur de PCI-SCEMM, de représenter notre partenaire industriel lors de cette soutenance de thèse.

Je remercie tous les membres du centre G2I et, plus particulièrement, l'équipe MSGI pour leur accueil et leur soutien. Merci également à Faicel, Jean-François et Marie Line pour leur aide et leur soutien.

Finalement, un grand merci à toute ma famille, ma bien aimée, mes amis ici en France et en ma Tunisie natale qui sont toujours présents pour moi quand j'en ai besoin.

TABLE DES MATIERES

LISTE DES FIGURES

LISTE DES TABLEAUX

LISTE DES ALGORITHMES

Introduction générale

Dans un contexte économique caractérisé par l'incertitude et une concurrence rude, les entreprises fabriquant des systèmes d'usinage se doivent d'être plus réactives et plus innovantes. En effet, les évolutions rapides de la demande rendent de plus en plus difficile la rentabilisation des investissements. Pour cela, ces entreprises s'intéressent à la réduction du cycle de conception de leurs produits, i.e., les lignes et systèmes d'usinage. La conception d'un système d'usinage prend ainsi une importance majeure. Son objectif principal est de proposer des architectures de systèmes d'usinage qui correspondent au mieux à la demande des clients dans les plus brefs délais, et qui soient moins chères et plus efficaces que celles proposées par la concurrence. Les systèmes d'usinage étudiés dans cette thèse sont conçus pour la production d'un seul type de produit. Néanmoins, un changement du produit ou de ses caractéristiques (techniques, géométriques, etc.) peut être constaté au cours de la phase de conception ou après l'installation et l'exploitation du système d'usinage. Dans ce cas, une voie prometteuse est de concevoir des lignes reconfigurables capables de s'adapter aux évolutions possibles de l'environnement commercial et industriel.

Cette thèse s'intéresse à la conception et l'optimisation de lignes de transfert reconfigurables. L'objectif principal est de concevoir une ligne d'usinage à moindre coût tout en respectant les contraintes techniques, technologiques et économiques. Les lignes de transfert sont utilisées dans la production de pièces pour l'industrie automobile, aéronautique, navale, etc. Elles nécessitent de lourds investissements en raison de coûts élevés d'installation, mais elles peuvent être rentabilisées grâce à un volume de production et une durée de vie importants qui permettent d'obtenir un coût par pièce réduit. Les solutions retenues à la conception de ces lignes influent donc directement et pour longtemps sur les coûts de production, ce qui explique l'intérêt d'une optimisation de la ligne dès l'étape la plus en amont de sa conception.

Nous nous intéressons ici à des lignes conçues pour donner la possibilité de changer rapidement et facilement le volume de production ou les caractéristiques du produit. Cette

reconfigurabilité est rendue possible grâce à l'utilisation de centres d'usinage à commande numérique. Le problème d'optimisation correspondant est un problème d'équilibrage de lignes d'usinage sujet à des contraintes spécifiques. Il consiste à affecter les opérations aux stations de travail en minimisant le coût de la ligne. En plus des contraintes habituelles de ce type de problème, à savoir, les contraintes de précédence, d'inclusion et d'exclusion, nous avons dû considérer des contraintes d'accessibilité. De plus, la spécificité principale des lignes reconfigurables par rapport aux lignes de transfert dédiées (qui utilisent des stations à boitiers multi-broches), vient du fait que les opérations de chaque centre d'usinage sont réalisées en série (centres mono-broches). Cette particularité rend souvent nécessaire la mise en place de stations équipées de plusieurs centres d'usinage travaillant en parallèle pour obtenir les volumes de production souhaités. Enfin, l'utilisation d'une tête d'usinage mono-broche induit la prise en compte de temps inter-opératoire de déplacements et de changement d'outils qui dépendent de la séquence d'opérations.

L'objectif de cette thèse consiste à développer des méthodes d'optimisation efficaces pour le problème d'équilibrage de ce type de lignes. Dans un premier temps, nous avons proposé une modélisation mathématique du problème à l'aide d'un programme linéaire en nombres entiers. Nous avons aussi développé des méthodes de calcul de bornes inférieures ainsi que des procédures de prétraitement. Cependant, les contraintes additionnelles rendent la résolution du problème d'équilibrage plus difficile que dans le cas des lignes dédiées, et l'approche proposée ne permet généralement pas de résoudre des instances de taille industrielle. Nous avons donc développé plusieurs méthodes de résolution approchées du problème en nous inspirant de métaheuristiques déjà utilisées avec succès sur d'autres problèmes d'optimisation combinatoire.

Dans le premier chapitre, nous nous intéressons aux problématiques liées à la conception des systèmes de production manufacturiers. Les définitions, les concepts et les termes nécessaires pour une bonne compréhension des différentes parties de ce mémoire y sont introduits. Nous décrivons aussi les systèmes de production d'une façon générale en évoquant leur évolution, leurs caractéristiques, les différents indicateurs de performances utilisés, etc. Les systèmes de production reconfigurables et les enjeux liés à leur utilisation sont aussi détaillés. Une attention particulière est apportée aux systèmes d'usinage.

Le chapitre 2 porte sur les problèmes d'équilibrage de lignes ainsi que sur les méthodes et approches dédiées à leur résolution. Nous commençons par présenter les principales

caractéristiques d'un problème proche largement étudié dans la littérature qui est celui d'équilibrage de lignes d'assemblage, ainsi que ses différentes variantes. Nous nous focalisons ensuite sur les problèmes d'équilibrage de lignes d'usinage ainsi que les problèmes d'équilibrage avec machines parallèles et ceux prenant en compte des temps inter-opératoires.

Le chapitre 3 décrit les principales caractéristiques du problème d'équilibrage étudié. Les lignes sont composées de centres d'usinage mono-broches qui peuvent être installés en parallèle sur la même station de travail. Une modélisation linéaire en nombres mixtes du problème est donnée. Nous proposons aussi une procédure de prétraitement pour faciliter la résolution du problème en réduisant sa taille.

Dans le chapitre 4, nous proposons différentes approches heuristiques pour la résolution du problème. Les approches développées sont basées sur un algorithme de construction utilisant des règles de priorité. Nous présentons trois approches basées respectivement sur COMSOAL, l'algorithme des colonies de fourmis et la méthode GRASP. Toutes les méthodes proposées sont testées sur un échantillon d'instances afin d'évaluer et de comparer leurs performances.

Dans le chapitre 5, nous proposons une illustration de l'approche générale sur un cas réel. Nous considérons ainsi un cas d'équilibrage de lignes pour l'usinage d'une culasse. Nous rapportons les solutions fournies par notre algorithme GRASP et nous analysons les principaux enseignements obtenus sur ce cas.

Chapitre 1
Conception des systèmes de production

Dans ce chapitre, nous nous intéressons aux problématiques liées à la conception des systèmes de production manufacturiers. Une attention particulière est apportée aux systèmes d'usinage. Nous introduisons les définitions, les concepts et les termes nécessaires pour une bonne compréhension des différentes parties de ce mémoire. Il existe plusieurs types de systèmes de productions. Ils peuvent être différenciés par leur configuration, architecture, système de pilotage, équipements utilisés, fonctions, etc. Les systèmes d'assemblage et d'usinage font partie des systèmes de production manufacturiers les plus utilisés dans l'industrie. Nous pouvons les classer en trois grandes familles, à savoir : les systèmes de production dédiés, les systèmes de production flexibles et les systèmes de production reconfigurables. Dans cette thèse, une attention particulière est portée aux systèmes de production reconfigurables.

Dans ce qui suit, nous introduisons les systèmes de production d'une façon générale en évoquant leurs évolutions, leurs caractéristiques, les différents indicateurs de performances, etc. Les systèmes de production reconfigurables et les enjeux liés à leur utilisation seront détaillés.

1.1 Systèmes de production manufacturiers

La production est un processus de transformation qui permet de convertir de la matière première en produits finis ayant de la valeur sur le marché. La transformation est effectuée par l'une ou une combinaison des opérations suivantes : travaux manuels, usinages, assemblages, etc. Le processus de transformation est composé généralement de plusieurs étapes (Groover, 1987). La production a connu plusieurs évolutions majeures depuis le milieu du $17^{ème}$ siècle. En effet, elle est passée d'un état primaire (agriculture, pèche, chasse, travail artisanal) à un état secondaire par l'introduction de la notion de manufacture avec des bâtiments et des

établissements entièrement consacrés à la fabrication et la transformation de produits grâce au travail manuel.

Au début du 20ème siècle, la production des biens manufacturiers a connu une évolution majeure par l'apparition des notions de marché et de commercialisation (Belhoste et al, 2004). Aussi, de nouvelles techniques et procédés de production ont été développés pour la fabrication de machines et d'outils. Ces avancées technologiques ont permis la transformation de la manufacture et la mise en place de systèmes de production mécanisés. Depuis, les systèmes de production ne cessent d'évoluer et de s'améliorer grâce aux nouvelles sources d'énergies, aux avancées technologiques, aux nouvelles méthodes de travail etc. Au milieu des années 60, la compétition entre les industriels devient plus intense. Cette compétition a été basée sur les prix. Plus les marchés deviennent complexes, plus les critères de qualité des produits et des délais de livraison deviennent importants. Ainsi, les entreprises se sont adaptées au nouvel environnement en attachant plus d'importance à la conception et à la commercialisation de leurs produits.

Un système de production manufacturier est composé de différents équipements : stations de travail, systèmes de manutention et de transport, systèmes de contrôle et de pilotage, etc. Les stations de travail sont composées en général d'une ou plusieurs machines (automatisées ou guidées) ou d'opérateurs humains qui effectuent les mêmes opérations suivant un cycle fixe. Les pièces en cours de fabrication dans le système sont appelées des encours. Des stocks tampons peuvent être installés en amont des stations de travail suivant la politique et le type de production. Ils ont pour rôle d'assurer le flux continu en cas de pannes machine ou dans le cas d'une station goulot dans le cadre d'une ligne de production. Une station goulot est une station qui a un temps de travail supérieur aux temps des autres stations. Les systèmes de manutention et de transport assurent le flux de pièces entre les stations de travail. Ils sont généralement composés de convoyeurs fixés sur des rails qui assurent le passage des pièces d'une station à une autre. Les pièces peuvent être chargées et déchargées des stations à l'aide de robots automatisés ou par des opérateurs alloués à cette tâche. Les outils de pilotage et de contrôle permettent en premier lieu d'assurer la réalisation des objectifs fixés du système de production, de contrôler et de corriger le fonctionnement en cas d'écart(s) observé(s) par rapports aux objectifs initiaux. Le pilotage et le contrôle d'un système de production est assuré grâce au flux d'information entre un centre de pilotage et les différentes composantes du système.

1.2 Les indicateurs de performance des systèmes de production

La performance d'un système de production peut être mesurée à l'aide de plusieurs indicateurs :

- Temps de cycle : il est utilisé quand le temps de production est limité avec un volume de production fixé. Il correspond à l'intervalle de temps qui sépare la sortie de deux produits finis. Notons que dans la littérature de gestion de production, il existe une autre définition du temps de cycle qui consiste au temps que passe la pièce dans le système de production lorsque celui-ci est sériel. Ces deux indicateurs sont complètement différents. Dans ce qui suit, nous retenons le premier indicateur, c'est-à-dire « takt time »;

$$T = \max_{k=1,..,m}\{T_k\} ;$$

où T_k est le temps de cycle local de la station *k*, m le nombre total de stations.

- Taux de production (cadence) : c'est le critère le plus utilisé, il correspond au nombre de pièces produites par unité de temps. Le temps de cycle est inversement proportionnel au taux de production, ainsi, si on veut augmenter la productivité, il faut diminuer le temps de cycle ;

$$t_p = \frac{1}{T} ;$$

- Coût unitaire estimé : il correspond au rapport entre le coût de production (coût des équipements, coût de fonctionnement, etc.) et le nombre total de pièces fabriquées pendant une durée déterminée. Ainsi, plus la productivité est élevée plus le coût unitaire est petit pour un coût global fixe. De même, pour une productivité fixe, plus le coût global est faible, plus le coût unitaire est faible ;

- Efficience : critère de l'utilisation des ressources du système. Elle correspond en général au résultat de la multiplication du temps de cycle par le coût total du système de production ;

$Eff = T * C$, avec C le coût total du système de production ;

- Flexibilité et reconfigurabilité : aptitude du système de production à faire face aux changements de produits et de cadences.

Durant la seconde moitié du 20ème siècle, les systèmes de production ont connu des évolutions majeures. Ainsi, les industriels ont dû faire évoluer leurs systèmes de production afin de répondre aux changements de leurs univers économiques. Plusieurs types de systèmes de production ont été développés. Ils varient suivant leurs configurations, leurs architectures, les équipements utilisés, le niveau d'automatisation, le type des systèmes de pilotage et de contrôle, etc. Pour une production de masse avec une demande certaine et bien défini, les industriels ont mis en place des systèmes de production dédiés (en anglais : Dedicated Manufacturing Systems, DMS). Les DMS permettent de produire le plus rapidement possible avec des coûts faibles. De nouvelles contraintes imposées par l'évolution des marchés vers plus de compétitivité et l'exigence des clients ont poussé les industriels à développer des systèmes de production flexibles (en anglais : Flexible Manufacturing Systems, FMS). Les FMS permettent de répondre à ces nouvelles exigences en proposant plusieurs types de produits d'une même famille en petite et moyenne quantité ainsi qu'une meilleure personnalisation de leurs offres (Kusiak, 1986). Les systèmes de production reconfigurables (en anglais : Reconfigurable Manufacturing Systems, RMS) ont été développés afin d'assurer une reconversion rapide du système de production dans le cas d'un changement de produit ou de cadence. Dans ce qui suit, nous présentons un aperçu général des DMS et des FMS et nous détaillons les caractéristiques, les différentes définitions et les principes de fonctionnement des RMS.

1.3 Systèmes d'usinage

Un système d'usinage est un système de production composé de plusieurs stations. Un ensemble prédéfini d'opérations d'usinage est exécuté sur chaque station afin d'obtenir un produit fini. L'usinage consiste à enlever de la matière de manière à donner à une pièce une forme voulue. Les systèmes d'usinage interviennent dans plusieurs domaines manufacturiers, tels que l'industrie automobile, l'aéronautique, la construction navale, etc. Il existe plusieurs procédés d'usinage, comme par exemple l'alésage, le fraisage ou le perçage. Une combinaison de ces opérations est généralement nécessaire pour la fabrication de pièces complexes (culasse, bloc moteur, etc., Figure 1.1). Des fonctions autres que l'usinage (assemblage, lavage, manutention, …) peuvent être exécutées dans un système d'usinage.

La composante principale d'un système d'usinage est la machine outil (un centre d'usinage à commande numérique ou une machine spéciale). Cette machine peut être spécifique ou polyvalente (capable d'effectuer plusieurs procédés d'usinage). Les premières machines-outils datent du début du 20ème siècle. Plusieurs types de machines-outils ont été développés. Ceci a permis de proposer des machines-outils avec différentes fonctions et caractéristiques selon les besoins des industriels. Dans cette section, nous détaillons les principales composantes et caractéristiques des systèmes d'usinage, à savoir, les opérations d'usinage, les centres d'usinage, les lignes et les systèmes d'usinage.

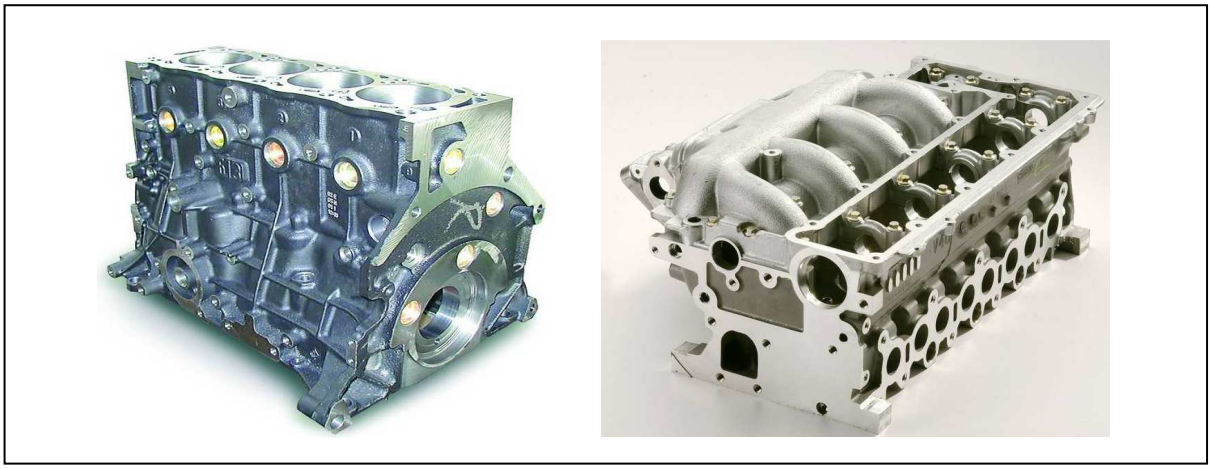

Figure 1.1 Exemples de pièces produites par des lignes d'usinage (PCI-SCEMM)

Le marché mondial de la machine-outil est dominé par les japonais, italiens, allemands et nord-américains. Des pays d'Europe de l'est et des pays émergents commencent à prendre une place importante dans ce secteur. Une concurrence acharnée est lancée pour gagner des parts de marché. La place de la France sur ce marché est modeste : elle assure 3% de la production mondiale et occupe le 10ème rang des pays producteurs de bien d'équipements. Ce secteur a connu des difficultés majeures durant la dernière décennie, néanmoins, des signes de reprise d'activité sont à noter avec une augmentation des exportations françaises en 2007 et 2008. Les principaux clients consommateurs sont la Chine et les pays d'Europe de l'est. Ceci s'explique par la délocalisation de secteurs consommateurs de machines outil, principalement le secteur automobile. En ce qui concerne l'Union Européenne, les principaux clients restent la Belgique, l'Italie, l'Allemagne et l'Espagne.

La France s'est engagée dans la production de machines de haute technologie afin d'affronter la concurrence. Ainsi, un grand effort a été apporté aux domaines de la recherche intervenants dans la fabrication de la machine outil, tels que les techniques d'usinage à haute vitesse, les

machines d'usinage multifonctions, les techniques de précision, etc. D'autres avancées notables ont été effectuées au niveau de la télémaintenance grâce à l'utilisation des nouvelles technologies de l'information.

Tableau 1.1 Commerce extérieur de la branche en France (Source INSEE)

Année	Exportations millions d'€	Importations millions d'€	Taux de couverture (%)
2000	1 540,4	3 386,6	45,5
2001	1 652,7	3 380,7	48,9
2002	1 499,5	2 693,3	55,7
2003	1 418,3	2 500,9	56,7
2004	1 405,6	2 669,5	52,7
2005	1 583,7	2 899,9	54,6
2006	1 670,6	2 881,5	58,0
2007	1 750,8	3 004,9	58,3

Tableau 1.2 Parts de marché des principaux pays exportateurs en 2006 (Source : Chelem – Cepii)

Pays	Part de marché (%)
Allemagne	19,4
Japon	16,5
Etats-Unis	10,9
Italie	8,4
Chine	8,1
Suisse	5,7
Taiwan	5,3
France	2,4
Corée du Sud	2,4
Royaume-Uni	2,3

1.3.1. Usinage

Comme nous l'avons mentionné précédemment, l'usinage consiste à enlever de la matière d'une pièce brute de manière à lui donner une forme voulue. L'usinage comporte un ensemble de techniques de fabrication. Dans la suite, nous détaillons les caractéristiques des principales techniques. Pour chaque technique, un outil d'usinage spécifique est utilisé. Deux méthodes d'usinage existent, à savoir, i) le travail de forme, la surface usinée est conditionnée grâce à la surface tranchante de l'outil d'usinage, ii) le travail d'enveloppe, il consiste à usiner la surface en utilisant la conjonction des mouvements de coupes et d'avance de l'outil d'usinage.

Pour l'usinage d'une pièce complexe, une gamme de fabrication est définie par le bureau des méthodes (gammiste). Cette gamme de fabrication est composée d'un ensemble d'opérations d'usinage mais elle peut aussi comporter des opérations d'assemblage ou de lavage. Ainsi, un gammiste définit l'ensemble des usinages, des techniques et des outils à utiliser en prenant en compte les dimensions, la géométrie, les tolérances d'usinage, le matériau de la pièce à usiner, etc. L'optimisation de la gamme d'usinage est primordiale pour minimiser les coûts de fabrication. En effet, l'usinage engendre plusieurs coûts liés au temps de travail, à la nature de la matière à enlever, à l'usure et à la maintenance de la machine. À ceci s'ajoutent les coûts des outils d'usinage, de l'énergie, etc.

L'usinage s'opère grâce au déplacement de l'outil d'usinage en interférence avec la pièce. Ce qui permet l'enlèvement de la matière en la transformant en copeaux. La conjonction de deux mouvements relatifs entre la pièce et l'outil sont nécessaires pour l'enlèvement de matière, à savoir : i) le mouvement de coupe ou vitesse de coupe défini en m/mn (vitesse de déplacement de l'arête tranchante de l'outil par rapport à la pièce) et ii) le mouvement d'avance ou vitesse d'avance (elle définit la quantité de matière enlevée). En général, plusieurs coupes successives sont nécessaires pour un seul usinage.

L'usinage présente plusieurs défis techniques. Une vitesse d'avance minimum est nécessaire pour assurer l'enlèvement de la matière (apparition des copeaux), elle dépend de l'état de la surface, du matériau à usiner et de l'outil utilisé. Si la vitesse d'avance est inférieure à la vitesse minimum, l'enlèvement de la matière est impossible et un écrouissage (modification de la structure interne de la matière) peut engendrer un échauffement préjudiciable à la pièce et à l'outil. La gestion des copeaux est importante pour le bon déroulement du procédé d'usinage. En effet, les copeaux formés sont mélangés à de l'huile ce qui forme de la boue qui doit être collectée et recyclée. Des procédés d'usinage ont été développés pour mieux gérer

les problèmes liés aux copeaux. Leur évacuation est facilitée par leur fragmentation en petites particules. Des courbes de fragmentation définissent la vitesse de coupe à appliquer en prenant en compte la forme de l'outil utilisé, le matériau et la géométrie de la surface à usiner (Dietrich, 1992).

Comme nous l'avons déjà évoqué, l'usinage engendre des coûts de fonctionnement. Ces coûts sont liés aux paramètres de coupe qui dépendent de l'outil utilisé, des caractères mécaniques et du matériau de la pièce à usiner. Ils sont donc définis indépendamment de la machine utilisée et des caractéristiques géométriques de la pièce et de l'outil. Les paramètres de coupe sont :

- La vitesse de coupe v_c (m/min) : vitesse de déplacement de l'outil par rapport à la pièce au point de contact ;
- L'avance par tour f (mm/tr) : distance usinée par tour, elle caractérise l'état de surface obtenu.

À partir des paramètres de coupe et des géométries de la pièce à usiner, le concepteur définit le type de machine-outil à utiliser ainsi que ses paramètres. Les paramètres machines sont :

- F_N : fréquence de rotation de la broche (tr/min) ;
- V_f : vitesse d'avance V_f (mm/min) ;

Pour chaque forme d'usinage, un outil et un procédé d'usinage spécifiques sont nécessaires. Les principaux procédés d'usinage sont :

- l'alésage : opération qui consiste à usiner une surface intérieure d'un cylindre (ou pièce creuse). C'est une opération de finition. Ainsi, l'alésage est utilisé pour avoir une meilleure précision dimensionnelle et une finition de l'état de surface amélioré. Les conditions de coupe (vitesse de rotation, avance, lubrification, matière usinée) doivent être prises en compte et optimisées pour réaliser des objectifs de précisions (Figure 1.2).

Figure 1.2 Outil d'alésage

- le brochage : il nécessite l'utilisation d'une broche (série de dents montées sur une pièce en acier). L'objectif du brochage consiste à usiner des trous circulaires afin de les agrandir et de les transformer en des formes non circulaires (carré, étoile, courbe, etc.). Le brochage peut également être utilisé sur la surface à l'extérieur de la pièce.

- le fraisage : c'est un procédé d'usinage de surface qui permet d'usiner différentes formes complexes. Il nécessite l'utilisation d'un outil (la fraise) et d'une machine-outil (la fraiseuse) (Figure 1.3). Des formes complexes peuvent être usinées grâce au fraisage en utilisant une machine à commande numérique.

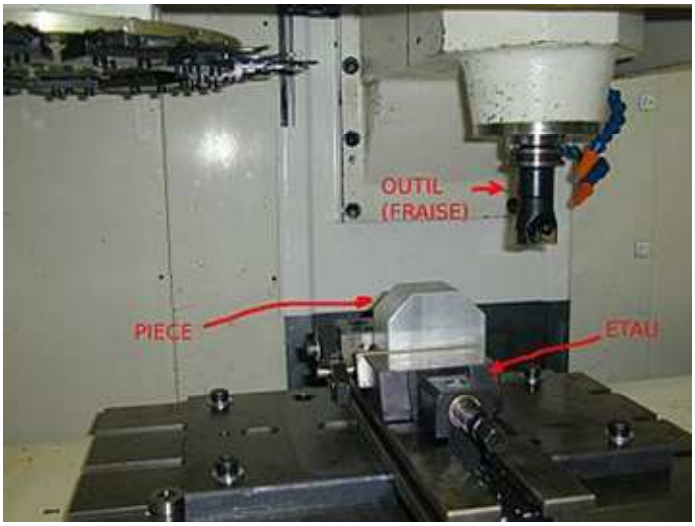

Figure 1.3 Opération de fraisage

- le lamage : c'est un perçage spécifique à font plat (aplanir une surface) afin d'assurer la portée de la tête d'une vis, un écrou, une rondelle, etc. Une fraise à lamer est utilisée pour ce type d'opération.

- le perçage : il consiste à percer un trou dans une pièce. Plusieurs outils peuvent être utilisés pour le perçage (foret (Figure 1.4), mèche, taraud), comme il peut être manuel ou mécanique. Le trou percé peut être simple ou taraudé suivant les fonctions recherchées (passer une pièce, un fluide ou une vis). Généralement, le perçage est utilisé comme opération d'ébauche, c'est-à-dire, préparer un trou pour une autre opération (taraudage, alésage, etc.). Dans le cas du perçage d'un trou de grande profondeur, des outils spéciaux doivent être utilisés. Le rapport diamètre longueur est important, dans le cas d'un rapport supérieur à 20 fois, on parle de forage. Ceci est imposé par la contrainte de l'évacuation des déchets dus au perçage (copeaux).

Figure 1.4 Outil de forage (3/4)

1.3.2. Machines d'usinage (machine-outil)

La machine-outil joue un rôle primordial dans les systèmes d'usinage. En effet, un grand effort a été apporté à la modernisation et la sophistication tant au niveau mécanique qu'au niveau de commande et de contrôle des machines outil depuis le début du 20ème siècle. En effet, le développement des micro-ordinateurs et micro-processeurs a permis la création des machines à commandes numériques (CNC machines). Le développement de nouvelles techniques d'usinage (usinage à grande vitesse (UGV), outil étagé, etc.) ainsi que l'amélioration des méthodes de conception (machine à axes multiples, machine avec boitiers multi-broches, etc.) ont permis d'avoir de nouvelles générations de machines outil avec de

meilleures performances. Dans la suite, nous présenterons les principales caractéristiques des machines-outils ainsi que les dernières avancées dans les domaines de leur conception et fabrication.

Une machine standard est composée de plusieurs éléments :

- Le bâti : il sert d'organe porteur des différents éléments de la machine et de la pièce à usiner. Il a un rôle primordial pour garantir la précision des usinages. En effet, le poids des éléments qu'il porte et les vibrations liées aux usinages peuvent causer des déformations statiques ce qui cause des erreurs de précision lors des usinages. Ceci implique de lui assurer une raideur statique ainsi qu'une rigidité dynamique suffisante afin de respecter les tolérances.

- La broche : elle permet de faire tourner l'outil lors de l'usinage. La broche est l'un des éléments les plus délicats lors de la réalisation d'une machine-outil. Ses fixations doivent être rigides et limiter les jeux afin d'assurer la précision et diminuer la consommation en énergie.

- Le porte-outil : il assure la fixation de l'outil sur la broche. Lors de l'usinage d'une pièce, différents outils peuvent être utilisés sur une même machine, dans ce cas, le changement automatique du couple outil/porte-outil est opéré quand c'est nécessaire (s'il y a deux opérations successives qui nécessitent deux outils différents). Ce couple (outil/porte-outil) constitue la tête d'usinage.

- L'outil : c'est l'instrument d'usinage. À chaque procédé d'usinage correspond un outil approprié. En général, dans le cas où des changements d'outil sont nécessaires, un magasin d'outils est installé dans la machine.

- Dispositif de fixation : la pièce est placée de façon à permettre aux outils d'effectuer les usinages nécessaires. Ce dispositif doit assurer la stabilité de la pièce afin qu'elle ne subisse pas de déformations statiques et d'assurer la qualité des usinages.

Les fonctionnalités d'une machine-outil dépendent du système de pilotage utilisé (manuel ou à commande numérique) et de ses caractéristiques mécaniques (degrés de liberté, taille, outils utilisé, etc.). Les degrés de liberté possibles sont les suivants :

- Trois translations selon trois axes (déplacement relatif de l'outil et de la pièce) ;
- Trois rotations autour de vecteurs parallèles à chaque axe (rotation de la broche qui porte l'outil ou rotation de la pièce) ;

La pièce doit être alignée avec les axes avec une précision supérieure ou égale à celle de tolérance la plus serrée afin d'éviter les défauts liés à l'alignement.

1.3.3. Les systèmes d'usinage

Il existe plusieurs types de systèmes d'usinage. Ils se différencient selon leur agencement physique. Nous pouvons distinguer trois types d'agencements physiques couramment utilisés, à savoir : agencement fonctionnel, agencement cellulaire et lignes de transfert.

- Agencement fonctionnel : les machines sont agencées et regroupées en sections homogènes et spécialisées par fonctions. Par exemple, une section pour le tournage, une autre pour le fraisage, etc. Les pièces passent d'une section à une autre suivant le processus d'usinage établi. Des allers et retours entre les sections sont possibles. Cet agencement permet d'avoir une meilleure flexibilité du système. Il est utilisé dans le cas d'une production diversifiée et en petits volumes. Néanmoins, certains inconvénients sont à noter : un coût de manutention élevé, une gestion de flux complexe, des encours importants, etc.

- Agencement cellulaire : le système est composé d'un ensemble de cellules qui assurent la production d'une famille de produits similaires. Ainsi, une cellule regroupe l'ensemble des processus, les hommes et les ressources nécessaires pour la production d'un ou de plusieurs produits de la même famille. Pour une production de petite et moyenne série, ce type d'agencement peut assurer une efficience de la production proche de celle de la production de masse. Néanmoins, il présente l'inconvénient que les ressources installées sont souvent sous-utilisées à cause du dédoublement des machines dans plusieurs cellules.

- Ligne de transfert : elle se compose généralement d'un ensemble de stations de travail agencées en série. Les stations sont ainsi visitées par les pièces suivant un ordre prédéfini. Le coût d'investissement de la ligne dépend du nombre des stations de travail et des équipements installés sur chacune des stations. Ces équipements sont définis à partir des opérations affectées aux stations. Généralement, chaque opération est caractérisée par : i) un temps opératoire, ii) un ensemble d'opérations qui doivent être exécutées avant (contraintes de précédence), iii) un ensemble d'opérations qui doivent être exécutées sur la même station de travail (contraintes d'inclusion), iv) un ensemble d'opérations qui ne peuvent pas être exécutées sur la même station (contraintes d'exclusion). Dans un contexte industriel, d'autres contraintes spécifiques peuvent être définies. En prenant compte de ces contraintes, à l'étape de configuration de la ligne, il est nécessaire de résoudre le problème d'équilibrage de la ligne. Ce problème consiste à affecter les opérations aux stations de travail afin de minimiser un ou plusieurs critères tout en respectant le taux de production.

Les lignes d'usinage en grande série (lignes de transfert) sont des systèmes de production largement utilisés dans les industries nécessitant l'usinage (Groover, 1987; Askin et Standridge, 1993; Hitomi, 1996; Dashchenko, 2003, 2006; Dolgui et Proth, 2006). Elles sont aussi utilisées dans les industries d'assemblage. Dans ce type de lignes, un ensemble répétitif d'opérations est exécuté à chaque temps de cycle. La ligne est composée de stations de travail qui sont à leur tour composées d'unités d'usinage. Les stations sont arrangées séquentiellement, et le système de manutention et de transport assure un flux constant de pièces entre les stations de travail. Le système de manutention se compose généralement des convoyeurs fixés sur des rails qui transfèrent les pièces d'une station à une autre à l'aide de robots automatisés ou d'opérateurs pour le chargement et le déchargement des pièces.

1.4 Principaux types de systèmes d'usinage

Il existe trois grandes familles de systèmes d'usinage à savoir : i) système d'usinage dédié pour une production de masse à long terme d'un seul type de produit, ii) système d'usinage flexible pour une production de plusieurs types de produits de la même famille en petite et moyenne quantité et iii) système d'usinage reconfigurable pour une production de moyenne et grande quantité d'un seul produit à moyen et long terme.

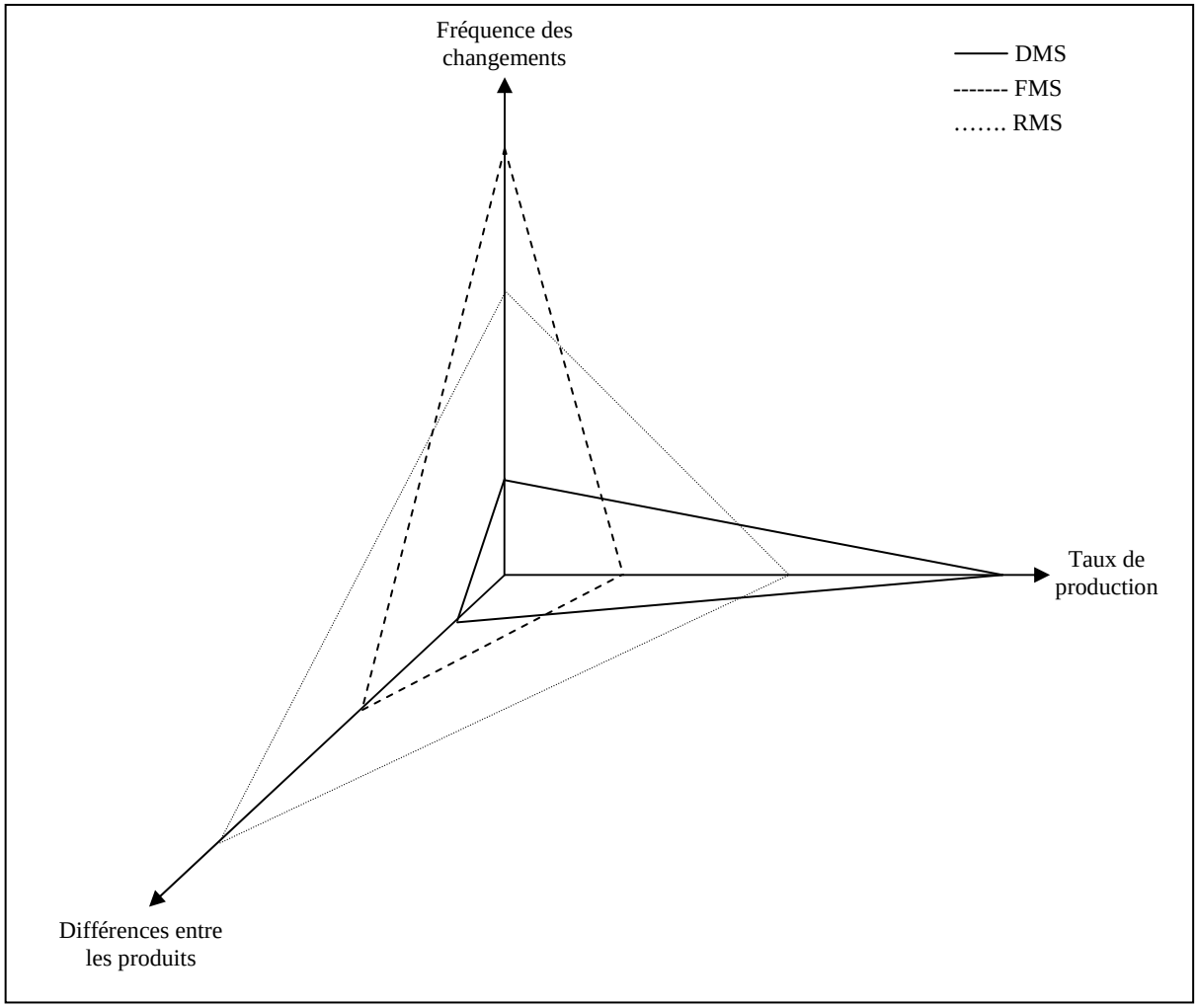

Figure 1.5 Performances des DMS, des FMS et des RMS

La Figure 1.5 illustre les différences en termes de fréquence des changements, différences entre les produits et taux de production entre les RMS, les FMS et les DMS. En effet, pour une production de masse avec un taux de production élevé, les DMS présentent le meilleur choix, tandis que les FMS sont conçus pour une haute fréquence de changement des produits. Les Systèmes de production reconfigurables (RMS) présentent un compromis entre les deux, c'est-à-dire, un taux de production et une fréquence de changements moyens avec plus de différences entre les produits. Dans la suite, nous détaillons les caractéristiques, les avantages et les inconvénients de chaque type de système.

1.4.1. Systèmes d'usinage dédiés

Les systèmes dédiés sont les plus économiques des systèmes d'usinage. Ils se caractérisent par une très grande productivité. Les DMS sont utilisés pour la production mono-produit en

grande série : une grande quantité de produits identiques est fabriquée avec la même séquence d'opérations sur les stations de travail.

Quand la demande du client est significative et stable à long terme, l'utilisation des DMS est la solution la plus profitable pour les industriels. Les principales caractéristiques de l'utilisation des DMS sont la réduction du temps de cycle et la minimisation du coût des équipements utilisés (machines, systèmes de transport, main d'œuvre). Ces coûts ont un effet direct sur le coût unitaire du produit final.

Résumons les principaux avantages des DMS :

- Production de masse : cela consiste à produire en large série un produit. Ainsi, les DMS permettent d'avoir une très grande productivité ce qui induit une réduction des prix de revient (Finel, 2004).

- Qualité : le haut degré d'automatisme des DMS et la standardisation des tâches de fabrication minimisent les risques de non-conformité de la qualité des produits.

- Durée de vie : les DMS sont en général conçus pour le long terme, ils peuvent fonctionner sur une période allant jusqu'à 30 ans avec des modifications à intervalles réguliers pour suivre les avancées technologiques ou pour intégrer des modifications au niveau du produit.

Les principaux inconvénients sont :

- Les DMS demandent un investissement conséquent et nécessitent une longue durée de vie pour être profitable. Le temps de reconversion de la production est relativement long.

- La prise en compte des caractéristiques spécifiques d'un seul produit durant l'étape de conception est possible, mais une fois le système de production défini, il devient très difficile de le changer (la reconfiguration de la ligne est coûteuse).

- Puisque les DMS sont généralement sériels, quand une station de travail tombe en panne, tout le système est bloqué. En plus, si une des opérations n'est pas terminée dans les temps à cause de cette panne, le produit est automatiquement défectueux.

Le critère principal à optimiser lors de la conception des DMS est facile à identifier et à calculer : minimisation du coût d'investissement. Néanmoins, l'optimisation des DMS lors de leur conception reste un problème difficile à résoudre. Donc, cette question reste d'actualité. En plus, la structure des DMS représente une forme basique pour l'organisation d'autres types de systèmes de production. En effet, tous les problèmes rencontrés lors de la conception d'un système de production dédié sont présents lors de la conception d'autres systèmes de production.

1.4.2. Systèmes d'usinage flexibles

Les FMS sont conçus généralement pour la production de plusieurs types de produits, appartenant à la même famille et afin d'assurer une personnalisation des produits. Par famille de produits, nous désignons les produits avec des dimensions comparables, des caractéristiques géométriques similaires, et des tolérances identiques. Les produits d'une même famille peuvent être fabriqués par les mêmes équipements. Les machines à commande numérique sont utilisées pour prendre en compte les changements en reprogrammant l'ordonnancement et le processus de fabrication. La flexibilité d'un FMS est également assurée grâce à l'utilisation de systèmes de transport et de systèmes de stockage automatisés avec des outils de contrôle sophistiqués.

Il existe trois types basiques de systèmes de production flexibles :

- Lignes flexibles : elles sont généralement composées de stations de travail arrangées séquentiellement. Elles peuvent assurer la production de différentes variantes d'un même produit avec des changements fréquents.

- Cellules flexibles : ces systèmes sont composés de machines à commande numérique interconnectées, où chaque machine exécute des processus qui incluent toutes les opérations nécessaire à la fabrication d'un seul produit. Le nombre des pièces de différents produits dans ce type de cellules est souvent restreint, de 8 à 10. Cela est dû à la capacité limitée des cellules.

- Systèmes flexibles : ils sont composés par des cellules flexibles liées. Ainsi, le processus de fabrication d'un produit peut être assuré par une ou plusieurs cellules. Il existe deux types de liaison entre les cellules : i) avec une séquence rigide de liaisons,

i.e, les cellules sont connectées avec un ordre invariable ; ii) avec des liaisons qui peuvent être adaptées à des produits particuliers.

L'objectif principal des systèmes de production flexibles c'est d'être capable de produire plusieurs variantes du même produit en petites séries. Ce type de systèmes de production assure un passage rapide d'une variante à une autre. Ces systèmes sont aussi capables de varier le volume de production d'un produit, si nécessaire, en diminuant le volume de production des autres produits. Néanmoins, les FMS présentent un certain nombre d'inconvénients (Koren et al., 1999) :

- Ces systèmes sont assez coûteux à cause des équipements utilisés. Ces équipements sont conçus pour une gamme de produits prévue et sont fabriqués sans un plan de production optimisé pour chaque produit. En effet, à l'étape de conception du système, les spécificités des gammes possibles de tous les produits à fabriquer sont prises en compte. Par conséquent, les concepteurs tendent à introduire des fonctionnalités qui peuvent être sous exploitées voire pas utilisées du tout. Cela augmente le coût total du système.

- Le développement des logiciels pour la commande des systèmes de production flexibles est aussi très coûteux.

- Confronté à la rapidité des avancés technologiques, les équipements et les logiciels sophistiqués et coûteux utilisés sont sujet à l'obsolescence.

- À cause de leur complexité, les systèmes de production flexibles sont peu fiables.

1.4.3. Systèmes d'usinage reconfigurables

Le concept de système de production reconfigurable a été introduit par Koren et al. (1999). Le développement de ce type de systèmes de production est motivé par les nouveaux enjeux imposés par les évolutions d'un marché de plus en plus mondialisé et incertain durant les années 90. La caractéristique majeure d'un RMS consiste à combiner une haute productivité des DMS et une agilité face aux changements comme des FMS. Un RMS est conçu afin d'être capable d'effectuer des changements rapides de sa structure, ainsi que de ses composants matériels et logiciels dans le but d'un ajustement rapide de la capacité de production et/ou de son adaptation à un nouveau produit ou une nouvelle famille de produits (Koren et al., 1999).

Un RMS peut avoir plusieurs niveaux de reconfigurabilité suivant les composants utilisés. Cette reconfigurabilité est assurée grâce à deux composantes reconfigurables principales, à savoir : une composante logiciel reconfigurable (système de pilotage et de contrôle à architecture ouverte) et une composante matériel reconfigurable (architecture, CNC machines, système de transport et de manutention). Cependant, ces deux composantes n'assurent pas toutes les conditions nécessaires pour un système de production réellement reconfigurable. En effet, le paradigme d'un RMS est une approche qui nécessite, en plus de la conception d'un système de pilotage reconfigurable et des ressources nécessaires, une méthodologie de conception systématique qui permettra en cas de reconfiguration, de réduire le temps et le coût de transition. Les caractéristiques et spécificités de ces composantes seront détaillées ci-dessous.

Ainsi, la structure d'un RMS est ajustable (réglable) ce qui permet une granularité de la production en réponse à une variation de la demande du marché. En plus de l'ajustabilité de la structure, elle est aussi adaptable à un changement éventuel du (ou des) produit(s).

Les principales caractéristiques des RMS sont :

- Modularité : dans un système reconfigurable, la plupart des composantes sont modulables (architecture, machines, logiciel, etc.). Ainsi, dans le cas d'une reconfiguration, le concepteur peut changer des modules rapidement et à moindre coût.

- Intégrabilité : afin de faciliter la conception des RMS, un ensemble de modules et leurs règles d'intégration doivent être établis.

- Personnalisation : cette caractéristique distingue les RMS des FMS et DMS, et peut réduire le coût total du système. En effet, la personnalisation d'un RMS a deux aspects distincts à savoir : la personnalisation du système physique et du système de contrôle. La conception des machines doit prendre en compte le produit (ou la famille de produits) à fabriquer pour assurer une personnalisation physique possible et une flexibilité et ainsi réduire le coût total du système. Le système de contrôle est conçu en intégrant les modules de contrôle nécessaires à l'aide d'une technologie à architecture ouverte ce qui permet d'apporter des modifications en cas de reconfiguration (en ajoutant, modifiant ou supprimant des modules de contrôle).

- Convertibilité : un changement rapide pour s'adapter aux produits appartenant à la même famille et une reconfiguration facile aux nouveaux produits.

- Diagnosticabilité : détecter les pannes machines et identifier les causes d'une qualité inacceptable des pièces. En cas de reconfiguration, ceci est très important pour réduire le temps de reconversion du système de production.

La diagnosticabilité, l'intégrabilité et la modularité permettent de réduire le temps de reconfiguration du système, tandis que la personnalisation et la convertibilité permettent de réduire le coût. Ainsi, ces caractéristiques peuvent donner une idée sur le niveau de reconfigurabilité d'un système de production.

La notion de système reconfigurable a été utilisée pour différents secteurs et domaines tels que les systèmes de production (cas étudié dans ce travail), systèmes informatiques, services, etc.

1.4.3.1. Enjeux

Comme nous l'avons déjà introduit, le développement des systèmes de production reconfigurables (RMS) a été imposé par les nouvelles exigences des marchés et par le constat que les systèmes de production flexibles ne satisfont pas complètement à ces exigences fortes. En effet, l'univers économique mondialisé se caractérise par une fréquence d'introduction des nouveaux produits, des changements au niveau des composants des produits existants, une fluctuation de la demande, des changements au niveau des régulations nationales (contraintes légales : critères environnementaux, droit du travail, etc.), et des changements rapides des processus technologiques. Pour faire face à ces changements et survivre dans un environnement industriel caractérisé par une concurrence rude, les manufacturiers doivent être capables d'adapter rapidement leurs systèmes de fabrication.

L'introduction de nouveaux produits a été accélérée par le développement d'outils logiciels d'aide à la conception (Computer-Aided Design, CAD). Néanmoins, quand le produit est un système de fabrication, il faut avoir des outils adaptés. Ce type d'outils logiciels commence à paraître dans tout ce qui s'appelle logiciel PLM (Product Life Cycle Management). Mais, la partie optimisation des systèmes de fabrication est encore embryonnaire dans ces logiciels, et donc dans la plupart des entreprises, le temps de conception de leur système de production reste relativement long. Ce constat implique que l'optimisation du processus de conception

des systèmes de fabrication et le développement d'outils logiciels correspondant est un problème d'actualité.

L'enjeu majeur des RMS concerne le développement de méthodes permettant une conception optimisée et rapide ainsi que les modèles et méthodes de reconfiguration pour une réadaptation du système en cas de changement au niveau du produit ou de la demande (volume de production). La reconfiguration d'un système de production nécessite l'intégration de nouvelles technologies (machines, capteurs) ainsi qu'une méthodologie de reconfiguration fiable qui prend en compte les caractéristiques et les spécificités du système en question. Réduire le temps de conception et de fabrication (ou de reconfiguration) est possible grâce à l'utilisation de composants modulaires au niveau des équipements et des outils de pilotage.

Les fluctuations des marchés nécessitent plusieurs types de reconfigurations suivant le changement en question. Ces changements peuvent nécessiter différents niveaux de reconfiguration. Cela peut aller d'une simple reconfiguration (ou remplacement) de machines dans le cas d'un changement au niveau du produit à l'ajout de machines pour accroître la productivité face à une augmentation de la demande. Durant le cycle de vie d'un système de production, plusieurs reconfigurations peuvent être effectuées. Ceci a comme conséquence, une durée de vie d'un RMS proche de celle des systèmes de production conventionnels (DMS), une optimisation de l'utilisation des ressources ainsi qu'une meilleure adaptabilité et réactivité de l'entreprise face aux aléas des marchés.

1.4.3.2. Système de pilotage et de contrôle

Un système de pilotage et de contrôle consiste en un logiciel qui permet de conduire différentes tâches à différents niveaux d'un système de production :

- Niveau bas : tâche de contrôle, de surveillance et de communication avec les composantes mécaniques, électriques et électroniques du système.

- Niveau supérieur : planification de la production, collecte et édition des données du système, contrôle du flux et de la qualité des produits.

Dans le cas des RMS, les logiciels de contrôle et de pilotage conventionnels ne sont pas adaptés. En effet, un RMS nécessite un système de contrôle modulaire, c'est-à-dire, un système composé d'entités séparées et découplées du reste du système ce qui rend un ajout ou

une modification d'un composant possible. Pour un RMS, le système de pilotage et de contrôle est une composante clef pour assurer une adaptabilité rapide du système de production en cas de changement du produit (ou famille de produits) ou de la quantité de la demande. La notion de système de contrôle à architecture ouverte a été retenue comme solution pour assurer le pilotage et le contrôle des RMS.

La programmation orientée objet (POO) a été adoptée comme outil qui peut apporter des solutions pour ce genre de système (Pritschow et Muller, 1997). La spécificité de la POO consiste en son utilisation comme application de contrôle temps réel (Awad et al., 1996), ainsi les différentes composantes d'un RMS sont pilotées et contrôlées en temps réel. L'outil logiciel utilisé ainsi que l'architecture du système de production doivent être compatibles et doivent interagir en harmonie pour satisfaire les caractéristiques essentielles d'un RMS.

1.4.3.3. Equipements

L'utilisation de machines et ressources reconfigurables est un des aspects importants des RMS. En effet, une technologie pour la conception systématique de machines reconfigurables et modulaires ainsi que des principes de conception d'un système de contrôle et de pilotage à architecture ouverte sont nécessaires pour le bon fonctionnement d'un RMS. La conception de ressources reconfigurables peut être basée sur une bibliothèque de modules qui peuvent être installés (ou retirés) sur une machine suivant les besoins du concepteur. Plusieurs types de ces machines ainsi que de méthodes de conception ont été développés pour différents systèmes de productions.

Dans le cas des systèmes d'usinage, une machines d'usinage reconfigurable peut effectuer différentes opérations telles que le fraisage, l'alésage, le perçage, etc. L'enjeu principal consiste à concevoir des systèmes d'usinage reconfigurables et optimaux en termes de coût, de productivité, de qualité produit ainsi qu'en temps de reconfiguration. La conception des machines modulaires est une composante clef pour assurer la reconfigurabilité d'un système de production. Ainsi, le système d'usinage pourrait être facilement reconfiguré par la suppression, l'ajout ou le changement des composantes ou des modules du système ou des machines.

La conception de machines d'usinage modulaires n'est pas une thématique nouvelle. Néanmoins, des avancées récentes ont ouvert de nouvelles possibilités pour la conception de ce type de machines. Ainsi, plusieurs types de machines outils ont été développés et mis au point avec comme objectif principal d'augmenter la reconfigurabilité tout en limitant les

coûts. Cette reconfigurabilité doit répondre aux changements au niveau des caractéristiques des produits à fabriquer, à savoir :

- Taille de la pièce : le changement de la taille des pièces à fabriquer peut être pris en compte en utilisant des modules (table de fixation, portes outil, etc.) avec différentes tailles. Ainsi, il suffit de changer les modules nécessaires avec les tailles correspondantes.

- Géométrie de la pièce : un changement de la géométrie de la pièce peut nécessiter des changements au niveau de la machine. Par exemple, si des opérations se trouvaient sur des faces inaccessibles, dans ce cas, la machine-outil doit être reconfigurée en ajoutant (ou en remplaçant) un ou plusieurs axes de rotation.

- Volume de production : la machine outil peut être adaptée en ajoutant un ou plusieurs porte-outil(s) dans le cas d'une augmentation du volume de production.

- Gamme opératoire : des changements au niveau de la gamme opératoire peuvent être enregistrés, ceci implique une réadaptation rapide de la machine. Cette réadaptation dépend exclusivement de la nature des changements (nouvelles opérations, nouvelles contraintes, etc.).

1.4.3.4. Architecture

Le choix de l'architecture est un des aspects fondamentaux des RMS. En effet, l'architecture d'un système de production influe directement sur sa reconfigurabilité ainsi que sur sa fiabilité et son coût. Plusieurs architectures existent, nous nous limitons ici aux lignes de production :

- Ligne sérielle : les stations sont installées en série (Figure 1.6), la pièce passe d'une station à une autre où un ensemble prédéfini d'opérations est exécuté. Ce type de ligne se caractérise par un bas coût d'installation et d'exploitation, mais il présente des inconvénients au niveau de la fiabilité (si une station tombe en panne, toute la ligne est bloquée).

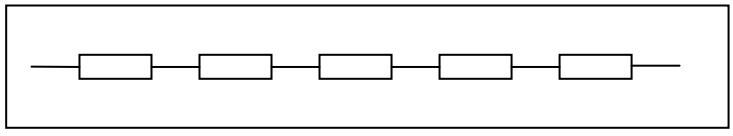

Figure 1.6 Ligne sérielle (la moins couteuse mais la moins fiable)

- Lignes parallèles : les mêmes lignes sont installées en parallèle (Figure 1.7). Chacune des lignes produit le même type de pièce. Ces systèmes sont plus coûteux mais présentent l'avantage de leur reconfigurabilité et se caractérisent par une meilleure fiabilité.

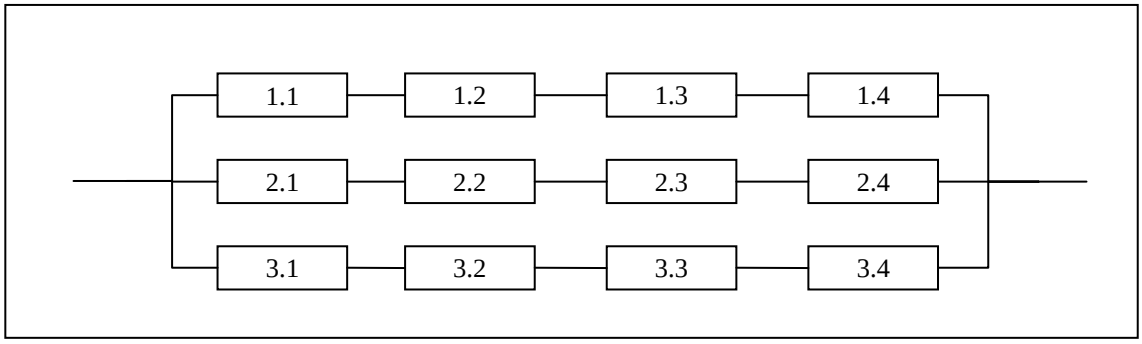

Figure 1.7 Lignes parallèles (3 lignes parallèles, 4 stations chacune)

- Ligne à stations parallèles (Figure 1.8) : ce type de lignes est un compromis entre les lignes sérielles et parallèles. En effet, ils présentent des avantages en termes de coût, de qualité des produits et de fiabilité.

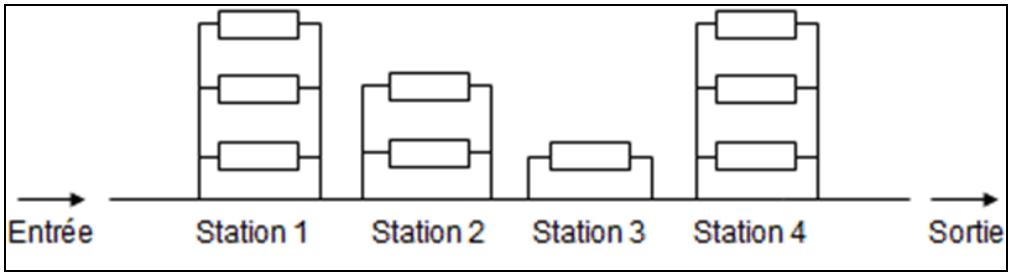

Figure 1.8 Ligne hybride (à stations parallèles)

1.5 Conception des lignes de transfert reconfigurables

1.5.1 Méthodologie générale de conception

Quelque soit le type de la ligne de transfert (dédiée, flexible ou reconfigurable), sa conception demande un ensemble d'approches qui nécessitent la résolution de plusieurs problèmes interconnectés (Askin et Standridge, 1993). Idéalement, les décisions en relation avec ces problèmes doivent être considérées simultanément. Cependant, le problème global est très complexe. Pour cette raison, il est nécessaire de décomposer le problème en plusieurs sous-problèmes, chacun nécessitant des prises de décisions moins complexes (Hitomi, 1996).

Nous précisons que nous ne considérons que l'étape de conception préliminaire (en avant projet), c'est à dire que certaines décisions structurantes concernant l'architecture du système de production et ses éléments sont prises. Généralement, ceci est suivi par une conception détaillée (spécification des éléments mécaniques, outils d'usinage, etc.) (Dashchenko, 2003, 2006).

Les étapes générales suivantes résument le processus de conception en avant projet. L'importance de chacune des étapes dépend du type de la ligne considérée.

- Analyse du produit: elle permet d'avoir une description complète des opérations nécessaires pour l'usinage du futur produit.

- Planification du processus d'usinage (choix de gamme) : elle concerne la sélection des processus d'usinage nécessaires pour transformer les pièces brutes en produit fini. À cette étape, les contraintes technologiques sont définies. Par exemple, durant cette étape, un ordre partiel entre les opérations ainsi que les contraintes d'inclusion et d'exclusion sont établis. Ceci nécessite une connaissance approfondie des spécifications fonctionnelles du produit et des contraintes technologiques liées aux procédés d'usinage.

- Problème d'équilibrage: cette étape consiste à résoudre le problème d'équilibrage, c'est-à-dire l'affectation des opérations aux stations de travail dans le but d'obtenir le taux de production nécessaire afin de satisfaire la demande tout en assurant la qualité recherchée. Il est impératif de considérer toutes les contraintes du problème, particulièrement, les contraintes de précédence, d'inclusion et d'exclusion.

- Analyse de flux et conception du système de manutention: La simulation est utilisée pour étudier le flux des produits en tenant en compte à la fois des évènements aléatoires externes et de la variabilité de la production. L'objectif consiste à analyser le flux dynamique, à choisir le système de manutention et à optimiser l'agencement physique, c'est à dire l'emplacement des machines et des stations de travail. Les décisions doivent être cohérentes avec celles des étapes précédentes.

- Conception détaillée et implémentation de la ligne.

Après l'implémentation et le début d'exploitation de la ligne, si le produit et/ou le volume de production changent, une analyse similaire doit être effectuée pour une reconfiguration optimale de la ligne de transfert (ceci est rarement considéré pour les lignes dédiées à la production de masse, plus précisément, la reconfiguration des lignes dédiées nécessite des approches spécifiques). Comme illustré dans la Figure 1.9, ces étapes sont exécutées séquentiellement. Le concepteur peut retourner aux étapes précédentes lorsque c'est nécessaire (le processus de décision est itératif).

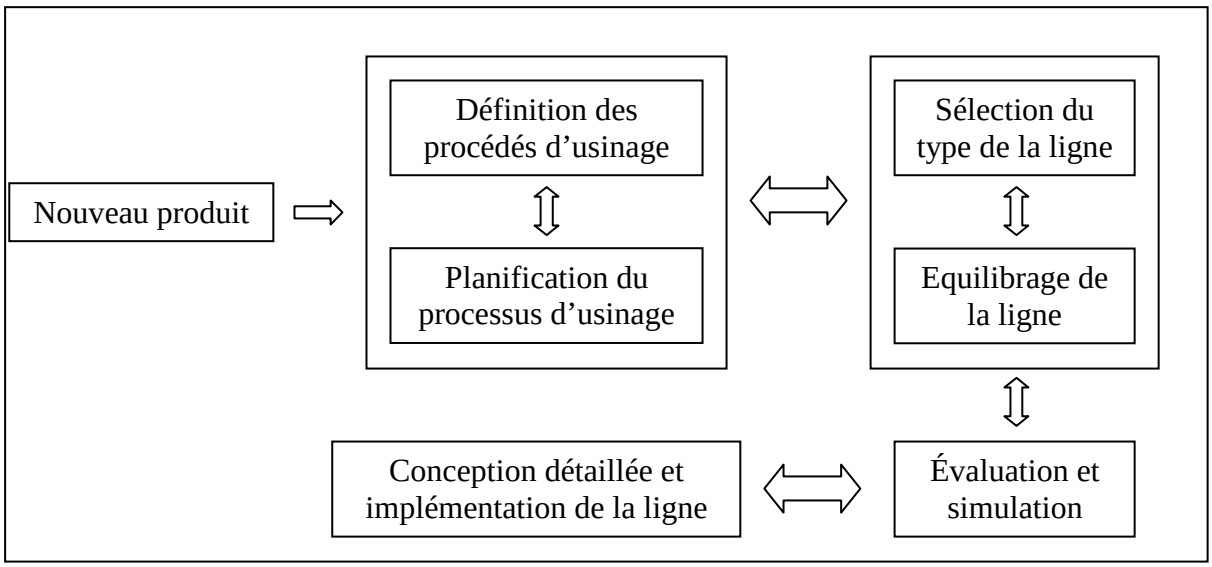

Figure 1.9 Processus de conception des lignes de transfert

Ce type de méthodologie a été déjà intégré dans un outil logiciel d'aide à la décision pour la conception préliminaire des lignes de transfert dédiées (Dolgui et al., 2009). Le logiciel développé inclut une base de données des éléments d'usinage (entités ou features en anglais). L'analyse du produit fournit un ensemble d'entités caractéristiques utilisées lors de l'étape de planification du processus d'usinage. La planification génère plusieurs gammes en combinant des gammes des entités d'usinage choisies. Un ensemble d'opérations et de contraintes d'usinage est obtenu. Après cela, le concepteur peut sélectionner le type de la ligne de transfert en considérant les différents processus d'usinage, les dimensions du produit, la productivité requise, le coût et les fonctionnalités des équipements, la variabilité de la demande, etc. Pour le processus d'usinage obtenu et le type de ligne sélectionné, le problème d'équilibrage correspondant est résolu ce qui donne une affectation des opérations aux stations et l'ensemble d'équipements nécessaires. Finalement, une estimation du coût du système de production ainsi obtenu est faite. Dans le cas où la solution ou le coût sont insatisfaisants, le concepteur peut modifier les données et les contraintes du problème et

relancer le processus. Cette approche est basée sur un ensemble de règles technologiques, une connaissance des contraintes de base et l'utilisation de techniques d'optimisation pour le problème d'équilibrage. Dans cette thèse, nous tentons d'apporter une contribution pour le développement d'un tel logiciel d'aide à la décision pour les RMS.

1.5.2 Problématique de l'équilibrage

Les lignes d'usinage étudiées sont équipées de centres d'usinage à commande numérique (CNC machines). Toutes les machines utilisées sont identiques. En contraste avec les lignes d'usinage dédiées (lignes d'usinage avec têtes multi-broches, Belmokhtar, 2006, Guschinskaya, 2007), les centres d'usinage comportent une seule broche et un magasin d'outil. La pièce est fixée sur un plateau rotatif qui permet l'accès à ses différentes faces. Néanmoins, quelque soit le type du plateau rotatif utilisé certaines faces restent inaccessibles (contraintes machines : axes de rotation). Ainsi, des contraintes d'accessibilités doivent être prises en compte, elles dépendent du plateau rotatif sur lequel la pièce est fixée (contrainte machine) et de la face d'appui de la pièce. Les machines utilisées sont mono-broche, ce qui implique de prendre en compte un temps additionnel entre deux opérations consécutives. En effet, dans une séquence d'opérations sur la même station, le passage d'une opération à une autre nécessite soit un changement d'outil soit, un déplacement d'outil ou la rotation de la pièce si les deux opérations ne concernent pas la même face.

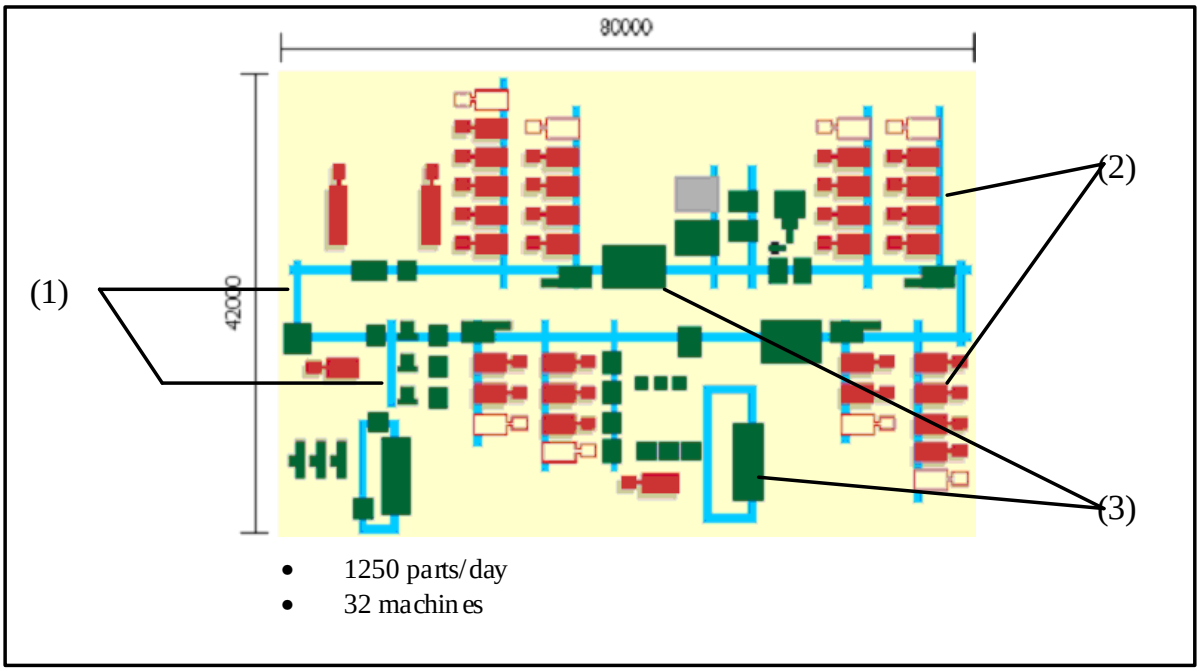

Figure 1.10 Configuration d'une ligne de transfert pour l'usinage de culasses (PCI-SCEMM)

La Figure 1.10 illustre l'architecture générale des lignes étudiées. Les stations sont arrangées en série et les pièces sont acheminées d'une station à l'autre grâce à un système de transport composé de convoyeurs ou d'une table roulante (Figure 1.10, lignes bleus, 1). Les pièces peuvent être chargées et déchargées des stations à l'aide d'opérateurs ou d'un système automatisé de manutention. Plusieurs centres d'usinage (CU) peuvent être installés en parallèle sur la même station de travail (Figure 1.10, en rouge, 2). Dans ce cas, chaque CU usine sa pièce en réalisant la même séquence d'opérations, c'est-à-dire les mêmes opérations sont dédoublées et exécutées suivant un ordre précis. Avec des machines parallèles installées sur la même station de travail, le temps de cycle propre à la station peut être modifié (dans la suite de ce texte nous expliquerons le fonctionnement d'une station de travail avec des machines parallèles). Des opérations de lavage ou d'assemblage sont exécutées sur des stations spécifiques (Figure 1.10, en vert, 3).

Le problème est caractérisé par un ensemble de données, à savoir : l'ensemble des opérations à affecter sur la ligne, les temps opératoires des opérations, les temps inter-opératoires entre les opérations (setup time) et les posages possibles (face d'appui et orientation de la pièce). Le problème d'équilibrage consiste à affecter les opérations aux stations de travail tout en respectant les contraintes du problème afin d'optimiser un ou plusieurs critères (coût de la ligne, nombre de machines, temps de cycle, etc.).

Dans la suite, nous supposons que les données suivantes sont connues :

- Ensemble de toutes les opérations qui doivent être exécutées sur la ligne.
- À chaque opération correspond un temps opératoire connu et fixe.
- Un temps inter-opératoire entre les opérations est pris en compte, il correspond au temps nécessaire pour le passage d'une opération i à une opération j. La prise en compte de ce temps est nécessaire lors d'un changement d'outil (deux opérations successives nécessitants deux différents outils d'usinage), d'un déplacement d'outil (les deux opérations usinent la même face et utilisent le même outil d'usinage).
- Contrainte d'accessibilité: ensemble des posages possibles sur les machines utilisées. Cet ensemble est déterminé à partir des caractéristiques mécaniques des machines ainsi qu'à partir des caractéristiques géométriques de la pièce à usiner.

- Temps de cycle imposé : il correspond au temps maximum écoulé entre la production de deux pièces successives. Dans une ligne synchrone, le temps de cycle est déterminé par la station de travail qui a le plus grand temps d'exécution auquel il faut ajouter le temps nécessaire pour déplacer la pièce d'une station à une autre.

- Contrainte de précédence : relation de précédence entre les opérations.

- Contrainte d'inclusion : ensemble d'opérations qui doivent être exécutées sur la même station.

- Contrainte d'exclusion : ensemble d'opérations qui ne peuvent pas être exécutées sur la même station.

1.6 Conclusion

Dans ce chapitre, nous avons présenté les systèmes de production de façon générale, puis nous avons abordé les systèmes d'usinage. Nous avons expliqué les différents types de systèmes d'usinage et nous nous sommes concentrés sur les systèmes de production reconfigurables. Nous avons introduit la problématique qui fait l'objet de notre étude et qui concerne la conception préliminaire en avant projet de lignes d'usinage reconfigurables. Ce problème a été très peu traité dans la littérature par rapport au nombre de travaux consacrés à la conception des lignes dédiées et flexibles. Nous tentons d'apporter dans cette thèse quelques éléments pour la modélisation et la résolution de ce type de problème. Dans le chapitre suivant, nous allons décrire les principales méthodes d'optimisation utilisées pour ce genre de problème ainsi qu'un état de l'art des travaux académiques proches de la problématique traitée dans cette thèse.

Chapitre 2
Méthodes d'optimisation et équilibrage de lignes

Dans le chapitre précédent, nous avons décrit les notions et principes essentiels pour la conception des systèmes de production, et plus particulièrement pour la conception des lignes d'usinage. Nous allons maintenant nous concentrer sur une étape clé de cette conception en nous intéressant aux méthodes et approches dédiées à la résolution des problèmes d'équilibrage de lignes. Le problème d'équilibrage a été introduit initialement pour la conception des lignes d'assemblage (Salveson, 1955). Nous commençons par présenter les principales caractéristiques du problème général d'équilibrage de lignes d'assemblage (Assembly Line Balancing Problem, ALBP) et de ses différentes variantes. Nous montrons ensuite les spécificités de l'équilibrage des lignes de transfert pour l'usinage en grande série.

2.1. Problème d'équilibrage de ligne

Dans cette section, nous introduisons le problème d'équilibrage des lignes d'assemblage et ses différentes variantes notamment le problème d'équilibrage des lignes de transfert. Le problème d'équilibrage des lignes d'assemblage a été largement étudié depuis la première publication le concernant (Salveson, 1955). Il consiste à affecter des opérations nécessaires à la fabrication d'un produit aux stations de travail d'une ligne d'assemblage en respectant les contraintes du problème (cadence, précédence, etc.). L'objectif est de minimiser le temps mort et donc le coût du produit fini. Le problème est souvent traité durant la phase de conception préliminaire d'une ligne d'assemblage ou lors d'un changement du produit ou de la demande. Une bonne analyse et un bon traitement du problème sont cruciaux pour le bon fonctionnement de la ligne ainsi que pour la compétitivité de l'entreprise (minimisation des coûts, respect des cadences, etc.).

2.1.1. Exemple de problème d'équilibrage

Dans le Tableau 2.1, nous présentons les données d'un problème d'équilibrage. Pour chaque opération, nous donnons son temps et les contraintes de précédence (Figure 2.1). Les opérations doivent être réparties sur les stations de travail de façon à respecter un temps de

cycle T_0 égal à 5 unités de temps (une unité de temps peut être : minute, seconde, etc.). Il peut y avoir un temps mort dans une station de travail, il correspond au temps d'inactivité par cycle. La charge d'une station correspond au temps durant lequel les opérations affectées à la station sont exécutées dans un cycle (temps opératoires, temps de chargement ou de déchargement, etc.).

Tableau 2.1 Données du problème

Opération	Temps opératoire	Prédécesseurs
1	3	-
2	1	1
3	2	1
4	1	2, 3
5	3	2, 3
6	3	5
7	1	5
8	1	7
9	3	8
10	2	9

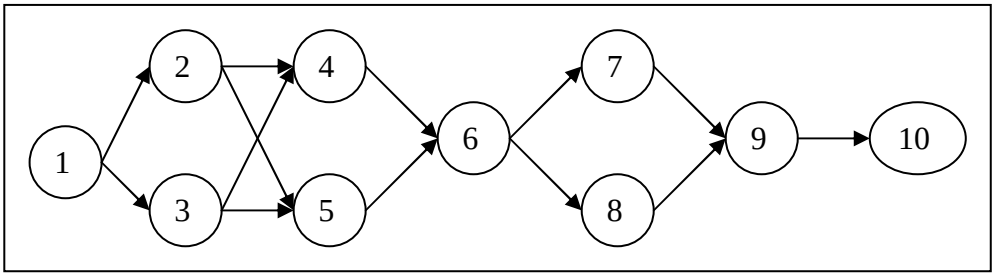

Figure 2.1 Graphe de précédence

Pour cet exemple, considérons deux solutions heuristiques à ce problème : une première solution consiste à affecter les opérations suivant leur ordre de numérotation (Figure 2.2, ligne 1) ; une deuxième solution peut être obtenue en classant les opérations suivant leur rang dans le graphe de précédence (Figure 2.2, ligne 2), les opérations ayant le même rang étant classées suivant l'ordre décroissant des temps opératoires. Lors de l'affectation des opérations, si une opération ne peut pas être affectée à une station (temps opératoire supérieur au temps libre de la station), nous passons à l'opération suivante, et ainsi de suite. Pour qu'une opération soit affectée à la station courante, il faut que les opérations liées avec elle par les contraintes de

précédence soient déjà affectées et que son temps opératoire soit inférieur au temps restant. Lorsqu'aucune opération ne peut être affectée à la station courante, nous créons une nouvelle station.

Les deux solutions précédentes sont admissibles, c'est-à-dire qu'elles respectent le temps de cycle et les contraintes de précédence. Néanmoins, la deuxième solution (ligne 2) est meilleure que la première (ligne 1) : elle comporte une station de moins et n'a pas de temps mort (temps de non utilisation des ressources) tandis que ligne 1 a 5 unités de temps mort par cycle. Cet exemple montre comment une affectation intelligente des opérations aux stations de travail peut générer des solutions de meilleure qualité que celles générées avec une affectation arbitraire. Malgré sa simplicité, il illustre ainsi l'importance de l'équilibrage qui permet entre autre de minimiser les coûts d'investissement et de fonctionnement (nombre de stations) en éliminant les gaspillages (temps morts).

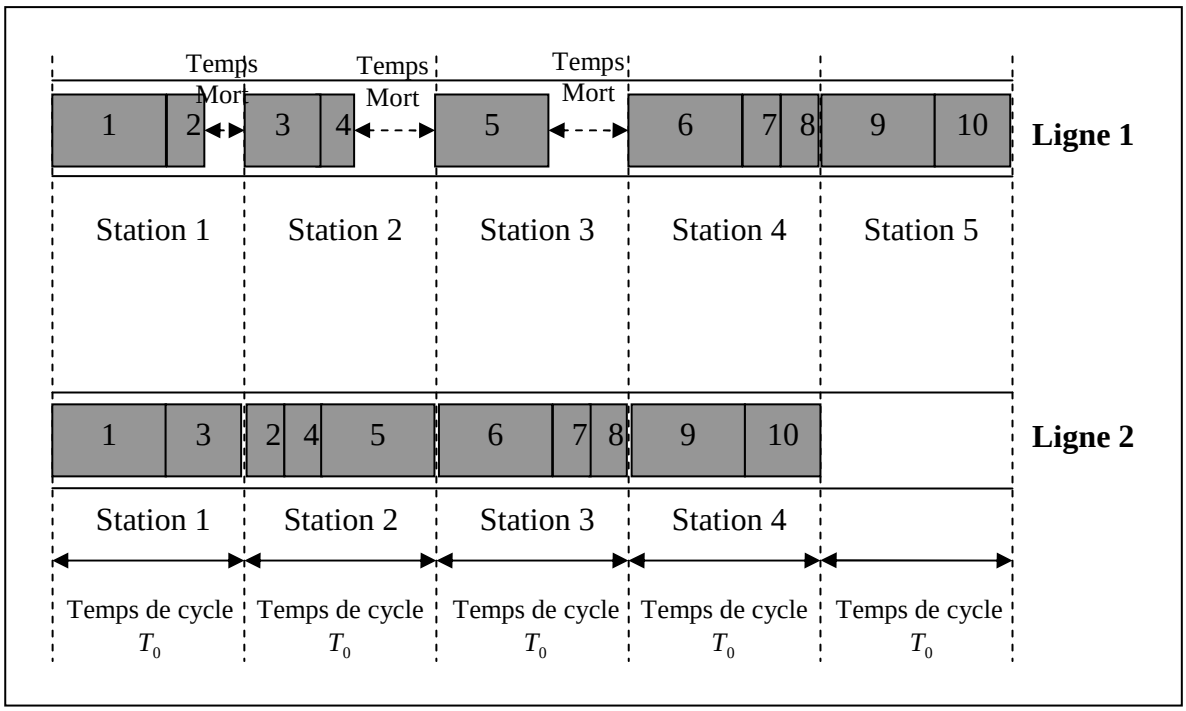

Figure 2.2 Exemple de deux solutions du problème d'équilibrage

2.1.2. Hypothèses et contraintes

Plusieurs états de l'art concernant les problèmes d'équilibrage des lignes d'assemblage ont été publiés (voir Baybars, 1986 ; Ghosh et Gagnon, 1989 ; Erel et Sarin, 1998 ; Scholl et al., 1999 ; Rekiek et al., 2002 ; Scholl et Becker, 2006 ; Boysen et al., 2007). Scholl (1999)

présente une classification des problèmes traités dans la littérature suivant différent critères (Figure 2.3), à savoir :

- le nombre de type des produits (mono-produit : simple, et multi-produit : mixte ou multiple) ;
- les caractéristiques des temps opératoires (stochastiques, déterministes, dynamiques) ;
- etc.

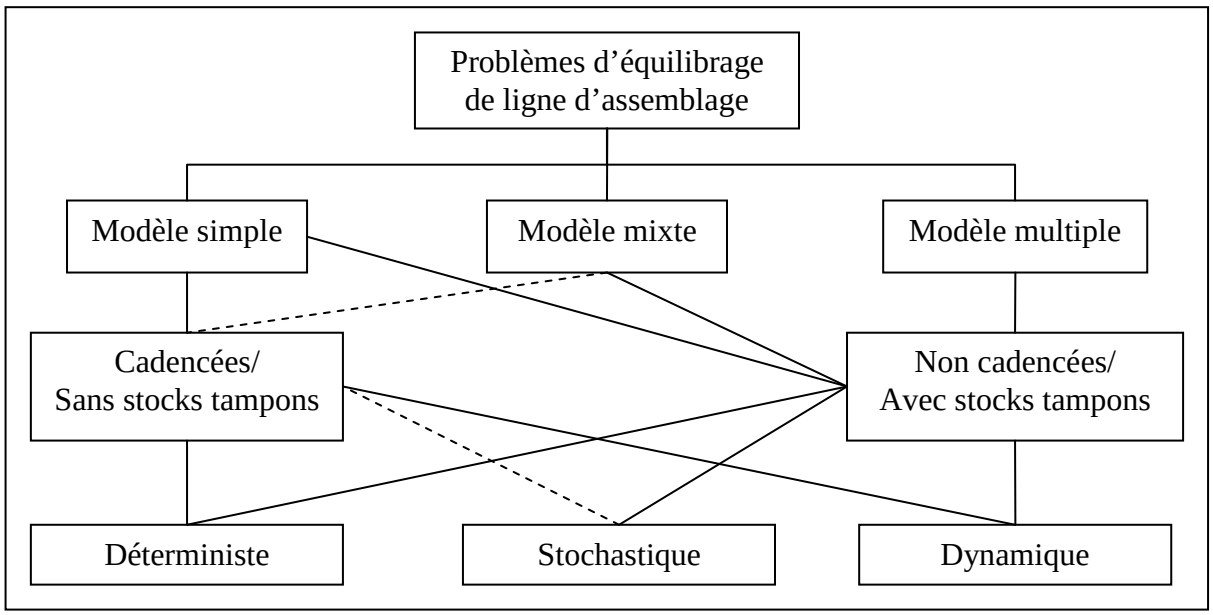

Figure 2.3 Classification des problèmes d'équilibrage

Dans son état de l'art, Baybars (1986) n'a considéré que les problèmes déterministes et les méthodes et algorithmes de résolution exacte. Il a ainsi distingué deux types de problèmes principaux, à savoir : le problème d'équilibrage des lignes d'assemblage simple (SALBP) et le problème d'équilibrage des lignes d'assemblage généralisé (GALBP). Il identifie les principales propriétés des problèmes :

- toutes les données du problème sont connues ;
- une opération doit être effectuée sur une et une seule station ;
- les contraintes entre les opérations doivent être respectées ;
- toutes les opérations doivent être effectuées.

En plus de ces propriétés, le SALBP se caractérise par les propriétés suivantes :

- n'importe quelle opération peut être affectée à n'importe quelle station ;
- les temps opératoires ne dépendent pas de la station ;
- n'importe quelle station peut exécuter n'importe quelle opération ;
- les stations sont disposées en série ;
- la ligne est conçue pour un seul type de produit et elle a un seul mode de fonctionnement.

À côté de ces hypothèses, des contraintes ou des données additionnelles peuvent être prises en compte pour des problèmes d'équilibrage spécifiques. Parmi ces données, nous pouvons citer :

- SALBP-1 : le temps de cycle est connu et fixe ;
- SALBP-2 : le nombre de stations est connu et fixe ;
- SALBP-F : le nombre de stations et le temps de cycle sont connus et fixes ;
- SALBP-E : le nombre de stations et le temps de cycle sont à minimiser ;
- Machines parallèles : une station peut être composée d'une ou plusieurs machines ;
- Temps inter-opératoire (temps de setup) : un temps additionnel entre deux opérations successives d'une séquence doit être pris en compte ;
- Opérations parallèles : des opérations différentes sont exécutées simultanément sur la même station de travail ;
- Contraintes d'inclusion : quand deux ou plusieurs opérations doivent être exécutées sur la même station ;
- Contraintes d'exclusion : pour deux opérations données, elles sont en exclusion si elles ne peuvent pas être affectées à la même station.

Le ALBP est un problème d'optimisation combinatoire NP-difficile (Bhattachajee et Sahu, 1990). Plusieurs méthodes d'optimisation peuvent être utilisées pour sa résolution. Dans la suite, nous nous intéressons aux problèmes d'optimisation combinatoire, à la notion de complexité ainsi qu'aux principales méthodes et approches de résolution développées dans la littérature.

2.2. Méthodes d'optimisation combinatoire

Les problèmes d'optimisation combinatoire correspondent habituellement à la recherche d'une solution optimisant (maximisation ou minimisation) un critère et satisfaisant un ensemble de contraintes. Une solution du problème se définit par des variables de décision

discrètes. Afin d'obtenir une solution, on cherche des combinaisons entre ces variables ce qui donne aux solutions possibles un caractère combinatoire. En effet, les valeurs de ces variables sont liées entre-elles par les contraintes : si une variable de décision change, elle influe sur une ou plusieurs autres variables. Pour les problèmes combinatoires, les relations entre les variables ont un effet considérable sur la difficulté de sa résolution. Deux classes principales de méthodes de résolution de ce type de problème se distinguent, à savoir : les méthodes exactes et les méthodes approchées. Avant d'aborder ces familles de méthodes, nous expliquons une théorie importante pour les problèmes combinatoires qui est la théorie de complexité.

2.2.1. Théorie de la complexité

Les problèmes d'optimisation combinatoire et les problèmes de décision qui leurs correspondent se caractérisent par leur complexité et la difficulté de leur résolution. Cette complexité est liée directement aux caractéristiques du problème, à savoir : la nature combinatoire du problème, le nombre des variables, les contraintes du problème et la fonction objectif à optimiser. Une manière naïve de trouver une solution optimale du problème, consiste à parcourir l'ensemble des solutions dans l'espace de recherche et de ne garder que la meilleure du point de vue de la fonction objectif.

La théorie de la complexité algorithmique a été conçue à l'origine afin de chercher si une réponse à un problème de décision (réponse oui ou non) peut être donnée efficacement. Elle peut aussi être appliquée au problème d'optimisation correspondant. L'efficacité de la recherche de solution permettra d'estimer approximativement le temps de calcul et le besoin en mémoire informatique nécessaires. Ainsi, des hiérarchies de difficultés des différents problèmes ont été mises en place par le biais de « classes de complexité ».

La complexité d'un algorithme est définie par le nombre d'instructions nécessaires pour la résolution du problème. Le choix de cet indicateur est motivé par le fait que le nombre d'instructions est indépendant de la puissance de la machine utilisée qui détermine le temps de calcul effectif. La complexité d'un algorithme est mesurée en grand $\mathcal{O}$ qui correspond à la complexité asymptotique dans le pire des cas. Elle est calculée comme un nombre maximum d'instructions élémentaires nécessaires pour résoudre n'importe quelle instance en fonction de sa taille n (où n est la taille du problème). La complexité d'un problème est différente de celle d'un algorithme utilisé pour sa résolution. Néanmoins, la complexité d'un problème peut être

évaluée comme celle du meilleur algorithme connu pour le résoudre (Paschos, 2005). Parmis les différentes classes de complexité, deux classes se distinguent : i) La classe P qui comporte les problèmes pour lesquels un algorithme de résolution de complexité polynomiale existe. Ces problèmes ont au moins un algorithme de résolution efficace (ou relativement efficace). ii) Les problèmes de la classe NP peuvent être décidés en temps polynomial avec des algorithmes non déterministes (le terme NP signifie Polynomiale Non déterministe). Un algorithme polynomial peut être utilisé pour tester la validité d'une solution du problème (Garey et Johnson, 1979).

De manière évidente la classe P est incluse dans NP mais la question de savoir si P est égale ou différente de NP est ouverte. Un problème de décision est dit NP-complet s'il appartient à la classe NP et qu'il est au moins aussi difficile que les autres problèmes de NP. Un problème d'optimisation est NP-difficile si le problème de décision associé est NP-complet.

Pour résoudre un problème NP-difficile, une explosion combinatoire est remarquée lorsque la taille des instances croît. Néanmoins, la complexité d'un algorithme est définie dans le pire des cas, il est donc possible de résoudre certaines instances d'un problème NP-difficile en un temps polynomial. Notons également qu'un algorithme polynomial ne garantit pas toujours l'absence d'explosion du temps de calcul car cela dépend du degré du polynôme et de la taille du problème.

2.2.2. Méthodes d'optimisation exactes

2.2.2.1. Programmation linéaire

Les méthodes de programmation linéaire (Linear Programming en anglais, LP) sont souvent utilisées pour une résolution exacte quand la fonction objectif et les contraintes du problème traité sont des fonctions linéaires des variables de décisions et que ces dernières sont des variables continues (Carlier et Chrétienne, 1988, Williams, 1993). Ces méthodes permettent de traiter des problèmes de grande taille grâce à leur efficacité (temps de calcul, nombre de variables généré, etc.). Dans un LP, la fonction objectif à maximiser ou à minimiser est définie comme une expression linéaire des variables de décision (2.1), quant aux contraintes du problème, elles sont définies à l'aide d'expressions linéaires bornées (2.2) :

$$\min z = cx \tag{2.1}$$

$$\text{Sous les contraintes } Ax \geq b; \tag{2.2}$$

$$x \in R_+^n \ ; \tag{2.3}$$

$$\text{avec } c \in R^n, b \in R^m, A \in R^{mn}$$

En interprétant le problème d'un point de vue géométrique, les contraintes linéaires forment un polyèdre convexe de l'espace euclidien R^n. Ce polyèdre correspond à l'espace des solutions réalisables *S*. L'espace des solutions est donc défini en fonction de ses facettes qui correspondent aux contraintes du programme linéaire et de ses points extrêmes définis par les intersections des contraintes. Dans le cas où l'espace S est non vide, le programme linéaire admet au moins une ou plusieurs solutions réalisables. La meilleure solution coïncide forcément avec un des points extrêmes de l'enveloppe convexe. Dantzig a développé en 1947 une méthode qui permet de parcourir les points extrêmes de l'espace afin de retrouver la solution optimale (voir, Dantzig, 1959, Cottle, 2003). Cette méthode est très utilisée pour la résolution des programmes linéaires et elle consiste à explorer les solutions voisines d'une solution initiale afin d'améliorer la fonction objectif. Une fois qu'aucune des solutions voisines n'est meilleure, l'optimum est atteint. L'algorithme de Dantzig n'est pas polynomial mais il a montré son extrême efficacité dans les applications réelles. Il existe des algorithmes polynomiaux pour la résolution des problèmes de programmation linéaire (Khachian, 1979, Karmarkar, 1984).

La programmation linéaire en nombres entiers (PLNE) est utilisée lorsque les variables de décision utilisées dans un programme linéaire sont toutes entières. De façon plus générale, les variables peuvent être de différents types, dans ce cas, on l'appelle programme linéaire en variables mixtes (Mixed Integer Program en anglais, MIP, voir Nemhauser et Wolsey, 1998). La relaxation LP d'un MIP donne une borne inférieure (pour le problème de minimisation) ou supérieure (pour le problème de maximisation).

2.2.2.2. Procédure par séparation et évaluation

La procédure par séparation et évaluation (PSE) est une méthode d'optimisation exacte qui consiste à énumérer implicitement les solutions en évitant de parcourir tout l'espace de recherche (Carlier, 1982 ; Carlier et Pinson, 1989 ; Demeulemeester et Herroelen, 1992, Dolgui et Proth, 2006). Ainsi, l'ensemble des solutions réalisables E est divisé en sous-

ensembles E_i divisés à leur tour en sous-ensembles, et ainsi de suite. Avec cette méthode, un arbre d'énumération multi-niveaux est construit, cela correspond à la séparation (Figure 2.4). Le premier niveau représente l'ensemble de départ, c'est le niveau racine. Les nœuds du niveau suivant sont créés par les nœuds du niveau précédent (nœuds ancêtres).

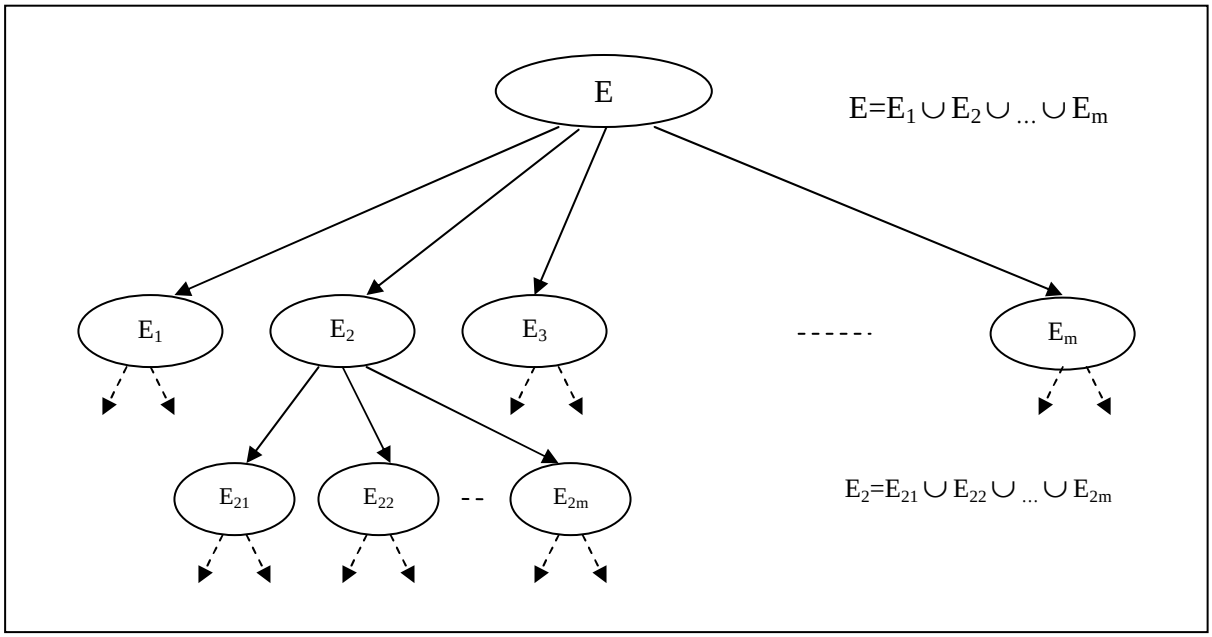

Figure 2.4 Arbre d'énumération

Afin d'éviter l'exploration de toutes les branches (réduire la complexité de l'algorithme), une évaluation est nécessaire. Ainsi, des bornes de la fonction objectif doivent être calculées à chaque nœud. Dans le cadre d'un problème de minimisation, une borne supérieure est calculée au début de l'énumération. L'exploration complète d'une branche de l'arbre permet de trouver une solution réalisable du problème (voire optimale). Au cas où une meilleure solution réalisable serait trouvée au cours du parcours de l'arbre (exploration d'une autre branche), elle devient la nouvelle borne supérieure. Une borne inférieure est calculée à chaque nœud exploré. Si au niveau d'un nœud, la borne inférieure est supérieure à la borne supérieure, le nœud et tous ses descendants sont supprimés (la valeur de la meilleure solution finale en explorant ce nœud sera moins bonne que la meilleure solution réalisable déjà obtenue). Cet arrêt permettra de ne pas explorer tout l'ensemble des solutions (ne pas parcourir tout l'arbre d'énumération) et ainsi d'accélérer la recherche. La qualité des bornes a une influence directe sur l'efficacité de l'approche (un meilleur calcul des bornes permet d'éliminer plus de branches). La recherche s'arrête une fois que toutes les bornes inférieures des nœuds à explorer sont supérieures ou égales à la borne supérieure de l'arbre. Cette

méthode est souvent utilisée pour trouver des solutions optimales des problèmes MIP en utilisant leur relaxation LP comme borne inférieure.

2.2.2.3. Autres méthodes d'optimisation exacte

Dans cette section, nous listons les principales autres méthodes exactes de résolution des problèmes d'optimisation combinatoire.

- Programmation dynamique : La programmation dynamique a été introduite par Bellman (1957). Elle se base sur le principe de l'optimalité (Bellman, 1957 ; Bellman et Dreyfus, 1962). Le problème initial est résolu à partir des solutions des sous-problèmes. La méthode commence d'un état initial x_0, à partir de cet état, un état suivant x_1 est déterminé à l'aide d'une décision u_1, ce processus est répété jusqu'à atteindre un état terminal x_n après la prise de décision u_n. L'ensemble des décisions $(u_1, u_2, ..., u_n)$ engendre un coût $f(u_1, u_2, .., u_n)$ (2.4).

$$f(u_1, u_2, .., u_n) = \sum_{i=1}^{n} f_i(x_{i-1}, u_i) \; ; \qquad (2.4)$$

- Programmation par contraintes : cette méthode permet d'utiliser les contraintes du problème dans le processus de recherche des solutions (Baptiste et al., 2003). En effet, le problème se caractérise par un ensemble de variables, chacune d'elle correspond à un domaine contenant toutes les valeurs que peut prendre cette variable. Un ensemble de contraintes lie les variables entre elles. Les contraintes sont ainsi utilisées pour réduire les domaines des variables à l'aide d'un mécanisme de déduction. La démarche de résolution utilisant la programmation par contraintes est effectuée à l'aide d'un algorithme de recherche arborescente. Cette démarche est effectuée en définissant une politique de branchement ainsi qu'une politique de retour en arrière (décisions à prendre en compte lorsqu'une incohérence est détectée).

2.2.3. Méthodes d'optimisation approchée

Les méthodes d'optimisation approchée sont développées afin de fournir des solutions de bonne qualité avec des temps de calcul raisonnables pour les problèmes complexes de grande taille. Les solutions fournies par ces méthodes ne sont pas nécessairement optimales mais des méthodes d'évaluation peuvent être utilisées pour avoir une idée de la qualité des solutions. Les méthodes constructives sont parmi les méthodes approchées les plus courantes. Elles

permettent de construire une solution progressivement à l'aide d'algorithmes utilisant par exemple des listes et tenant en compte de règles heuristiques (règles de priorité). Ce type de méthode heuristique, appelée algorithme glouton, permet de fournir une solution rapidement mais de qualité généralement perfectible. Certaines heuristiques peuvent être beaucoup plus sophistiquées et fournir des résultats de meilleure qualité. Outre qu'elles ne garantissent pas l'optimalité, le principal inconvénient de ces méthodes est qu'elles sont dédiées à un problème. Des méthodes de résolution plus générales (métaheuristiques) ont été développées avec un caractère indépendant des spécificités des problèmes traités. Dans Osman et Laporte (1996), les auteurs définissent une métaheuristique comme étant un processus itératif de génération dirigeant une heuristique subordonnée et combinant plusieurs techniques afin d'explorer au mieux l'espace de recherche dans le but d'atteindre des solutions proches de l'optimum. Notons que les métaheuristiques ne garantissent pas non plus l'optimalité.

Dans le paragraphe suivant, nous allons présenter les principales méthodes et algorithmes utilisées pour la résolution approchée des problèmes d'optimisation combinatoires.

2.2.3.1. Méthodes par voisinage

Les méthodes par voisinage se basent sur une génération itérative de solutions voisines. Le voisinage d'une solution correspond aux solutions obtenues en effectuant un seul mouvement à partir de la solution initiale (par exemple, permuter deux opérations dans un problème d'équilibrage). Une recherche par voisinage est une succession de ces mouvements. En général, ce type de méthodes est composé d'un algorithme qui génère des solutions admissibles et d'une méthode de recherche dans le voisinage des solutions générées. Plusieurs algorithmes et méthodes utilisent cette technique, les plus connues sont : le GRASP (Greedy Randomized Adaptive Search Procedure), le recuit simulé et la recherche tabou. Pour ces différentes méthodes, plusieurs stratégies de recherche sont appliquées. Nous présentons un aperçu général du fonctionnement de ces trois méthodes.

Le GRASP (Feo et Resende, 1989 ; Resende et Ribeiro, 2006) se base sur un algorithme semi-glouton et une méthode de recherche par voisinage utilisant un algorithme de descente (Figure 2.5). Cette méthode sera détaillée plus en détail dans la suite du mémoire. La recherche tabou s'appuie sur une liste tabou qui inclut tout les voisins visités afin d'éviter de visiter plusieurs fois le même. La définition du voisinage et le réglage des paramètres utilisés influent sur la qualité de la solution finale.

Le recuit simulé utilise une stratégie qui accepte une dégradation de la qualité d'une solution afin de s'extraire d'un optimum local (Kirkpatrick et al., 1983). Ce principe de dégradation est inspiré de l'opération de recuit (annealing en anglais) observée dans la sidérurgie et dans l'industrie du verre. En effet, après avoir subi des déformations, le métal est réchauffé à une certaine température (la structure atomique du métal est dégradée) puis refroidi lentement. Il prend alors une structure stable et acquière de meilleures propriétés. Ce principe s'applique facilement dans un algorithme de recherche par voisinage. Ainsi, l'algorithme de descente est modifié afin de permettre une dégradation de la solution. Cette dégradation est soumise à une probabilité qui dépend de l'ampleur de la dégradation ainsi que d'autres paramètres. Au fur et à mesure de l'avancement de l'algorithme, cette probabilité décroit jusqu'à devenir nulle.

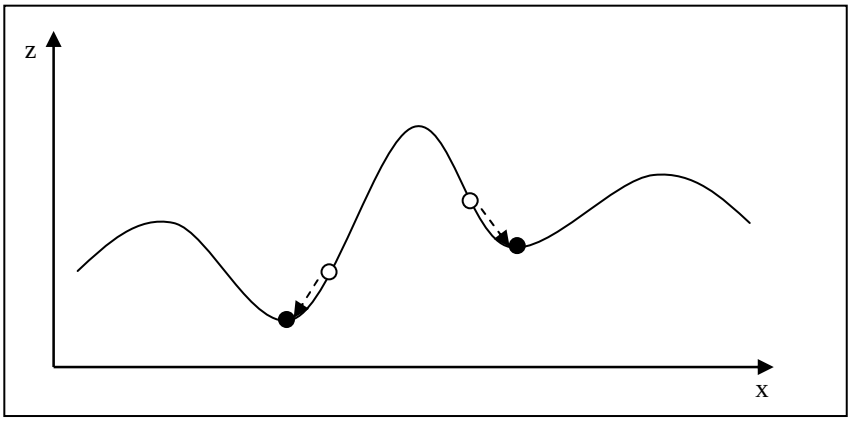

Figure 2.5 Principe de fonctionnement d'un algorithme de descente

2.2.3.2. Les algorithmes à base de population

Plusieurs algorithmes à base de population générant un ensemble de solutions simultanément ont été développés pour la résolution des problèmes d'optimisation difficiles en s'inspirant de la génétique ou de la biologie. Nous citons deux des principales méthodes, à savoir : les algorithmes génétiques et les algorithmes de colonies de fourmis :

- Les algorithmes génétiques s'inspirent de la biologie et de la génétique (comportement des chromosomes, individus) en utilisant des techniques de croisement et de mutation afin d'assurer la génération de nouvelles solutions. Plusieurs points doivent être relevés afin d'assurer un bon fonctionnement de l'algorithme, à savoir : bien coder les solutions (les parents), développer de bonnes méthodes de recombinaison et de mutation afin d'assurer la bonne qualité des solutions générées (les descendants), bien traiter et réparer les solutions non réalisables générées.

- Les algorithmes de colonies de fourmis s'inspirent des comportements collectifs de dépôt et de suivi de piste observés dans les colonies de fourmis. Ces colonies sont composées par des agents (les fourmis) qui communiquent indirectement via des modifications dynamiques de leur environnement (pistes de phéromones) et construisent une solution à un problème en s'appuyant sur leur expérience collective.

2.3. Équilibrage de lignes d'assemblage

Nous allons maintenant nous intéresser plus spécifiquement aux problèmes et méthodes liés à l'équilibrage des lignes de transfert étudiés dans ce mémoire. Le problème d'équilibrage des lignes de transfert est un cas particulier du problème d'équilibrage de ligne d'assemblage, puisqu'il présente les principales caractéristiques de ce dernier. Selon (Bhattachajee et Sahu, 1990), le SALBP est un problème d'optimisation NP-difficile. Plusieurs travaux de recherche ont été effectués pour la résolution du problème SALBP par le développement de techniques et de méthodes de résolution (Helgeson et Birnie, 1961 ; Moodi et Young, 1965 ; Pinto et al., 1975 ; Boctor, 1995).

Les modèles mixtes présentent une difficulté supplémentaire car ils s'intéressent à l'équilibrage des lignes avec plusieurs types de produits. Pour chaque type, ses propres contraintes et données doivent être prises en compte. À chaque instant, un mélange de produits de différents types est présent sur la ligne. La production respecte un ratio entre les types de produits. Le modèle multiple, comme le modèle mixte, concerne la production de plusieurs types de produit. Sauf que, la production des différents types de produits est lancée par lots. Ainsi, ce modèle présente non seulement le problème d'équilibrage de chaque lot, mais aussi l'ordonnancement du lancement de ces lots.

La plupart des modèles traités considèrent des temps opératoires fixes et connus. Sauf que des variations sur ces temps peuvent être engendrées, par exemple par une instabilité des performances des opérateurs humains. Ainsi, des temps opératoires aléatoires ou flous peuvent être considérés dans le cas des lignes manuelles. Peu de travaux ont été consacrés à ce types de lignes en comparaison avec les modèles déterministes (Suresh et Sahu, 1994 ; Erel et Sarin, 1998 ; Dolgui et Proth, 2010).

Dans la section suivante, nous présentons quelques méthodes utilisées pour la résolution des problèmes d'équilibrage des lignes d'assemblage.

2.3.1. Méthodes exactes

2.3.1.1. Modélisation linéaire du problème

Plusieurs modélisations du problème SALBP ont été présentées dans la littérature (Bowman, 1960 ; White, 1961 ; Patterson et Albracht, 1975 ; Scholl, 1999). White (1986) propose une modélisation linéaire du problème simple pour SALBP-1 développée à partir du modèle de Bowman (1960). Les variables de décision sont des variables binaires et correspondent à l'affectation des opérations aux stations :

$$x_{ij} = \begin{cases} 1 & si\ l'opération\ i\ est\ affectée\ à\ la\ station\ j \\ 0 & si\ non \end{cases}, pour\, i \in N\, et\, j \in M \ ;$$

$$c_j\ coût\, d'une\, station\, j, \forall j \in M \ ;$$

Avec $N = \{1,2,..,n\}$ l'ensemble des opérations à affecter et $M = \{1,2,..,m\}$ l'ensemble des stations disponibles. Soit P_i l'ensemble des opérations qui sont liées par des contraintes de précédence avec l'opération *i* (sont ses prédécesseurs). Nous avons donc un modèle linéaire en variables booléennes.

La fonction objectif est définie de la façon suivante :

Minimiser $F(x) = \sum_{i \in N} \sum_{j \in M} c_j x_{ij}$;

Avec $c_{j+1} \geq c_j, \forall\, j \in \{1,2,..,m-1\}$;

Sous les contraintes :

$$\sum_{j=1}^{m} x_{ij} = 1, \forall i \in N \ ;$$

$$\sum_{i=1}^{n} t_i x_{ij} \leq T, \forall j \in M \ ;$$

$$x_{ik} \leq \sum_{j=1}^{k} x_{hj}, \forall i \in N, \forall k \in M, \forall h \in P_i \ ;$$

$$x_{ij} \in \{0,1\}, \forall i \in N, \forall j \in M \ ;$$

Un autre modèle est présenté par Scholl (1999) qui propose une modélisation du problème en variables mixtes en combinant le modèle de Bowman (1960) et Patterson (1984). L'auteur utilise une variable entière z_i qui représente le numéro de la station à laquelle une opération i est affectée ainsi qu'une variable réel y_i qui correspond à la date de début d'une opération i à chaque cycle. Les opérations sont numérotées suivant un ordre croissant correspondant à leur rang dans le graphe de précédence.

2.3.1.2. Fable

C'est une procédure par séparation et évaluation dont les branchements se font par l'ajout d'une opération choisie parmi les candidats à la station courante. Ainsi, l'énumération est orientée opération. Si au cours de l'énumération aucune opération ne peut être affectée à la station courante, cette station est fermée et une nouvelle station est créée (elle devient à son tour la station courante). Une solution admissible n'est obtenue qu'une fois toutes les opérations affectées. Tant qu'il reste des branchements ouverts, des retours en arrières sont opérés. Des nœuds peuvent être supprimés s'ils correspondent à des solutions partielles qui contiennent le même ensemble d'opérations. Des règles de dominances sont utilisées afin de réduire d'avantage le nombre d'énumérations (branchements). Parmi ces règles, celle proposée par Jackson (1956) stipule qu'une opération i est dominée par une opération j si le temps opératoire de i est inférieure à celui de j et que l'ensemble des successeurs de i est inclus dans celui de j.

2.3.1.3. Eureka

Contrairement à l'algorithme Fable, Eureka est basé sur une énumération orientée station. Ainsi, à chaque branchement, une recherche est effectuée pour construire des combinaisons d'opérations réalisables formant une station. Une combinaison est choisie et affectée à la nouvelle station ce qui forme un nouveau nœud de l'arbre d'énumération. Cette méthode utilise une technique d'incrémentation de la borne inférieure (voir, Dolgui et Pashkevich, 2001).

2.3.2. Méthodes approchées

2.3.2.1. Heuristique

Différentes heuristiques ont été proposées pour la résolution du problème d'équilibrage des lignes d'assemblage (voir par exemple : Helgeson et Birnie, 1961 ; Dar-El-Mansoor, 1964 ;

Arcus, 1966 ; Talbot et al., 1986 ; Boctor, 1995 ; Ponnambalam et al, 1999). Nous présentons brièvement les heuristiques les plus connues.

RPW (Ranked Potitioned Weight) est une heuristique proposée par Helgeson et Birnie (1961). Elle consiste à affecter les opérations suivant une règle de priorité basée sur la somme des temps opératoires de l'opération ainsi que ceux de ses successeurs. Ainsi, les opérations sont ordonnées en ordre décroissant de leur poids. Une opération n'est affectée que si tous ses prédécesseurs ont été affectés à leur tour. Si la station est chargée et aucune opération ne peut lui être affectée, elle est fermée et une nouvelle station est ouverte. Cette méthode a fait ses preuves pour des problèmes simples.

COMSOAL (COmputer Method of Sequencing Operations for Assembly lines) est une méthode plus évoluée qui a été proposée par Arcus (1965). Elle consiste à générer plusieurs solutions et à récupérer la meilleure d'entre-elles. La méthode commence par créer une première station, elle devient la station courante. Une liste d'opérations qui peuvent être affectées à la station courante (contraintes de précédence, temps libre de la station courante, etc.) est créée. Une opération de cette liste est sélectionnée au hasard et affectée à la station courante. Si la liste est vide et qu'il existe toujours des opérations non affectées, une nouvelle station est créée et la liste est réinitialisée, sinon l'algorithme est terminé. À chaque affectation, la liste d'opérations est mise à jour. Cet algorithme est répété jusqu'à une condition d'arrêt (nombre maximal d'itérations, temps maximal, etc.). Une fois le calcul terminé, seule la meilleure solution est retenue.

2.3.2.2. Métaheuristique

Les métaheuristiques sont utilisées pour l'équilibrage des lignes d'assemblage comme méthodes complémentaires. Elles permettent entre autre d'améliorer la qualité des solutions obtenues par les heuristiques et de pallier aux limites des méthodes exactes pour les cas complexes (grande taille, limite de temps de calcul, etc.). Les méthodes les plus utilisées sont les algorithmes génétiques (Jones et Beltramo, 1991 ; Rubinovitz et Levitin, 1995 ; Rekiek et al., 2000 ; et Kim et al., 2001), le recuit simulé (Surech et Sahu, 1994 ; McMullen et Frazier, 1998) et la recherche tabou (Seriano et Gendreau, 1997 ; Wen-Chyan, 1998 ; Lapierre et al., 2006).

Ces méthodes se basent sur une méthode heuristique afin de générer un ensemble de solutions initiales et tentent d'apporter des améliorations. La qualité du résultat n'est pas garantie par de

telles méthodes, et la plupart des auteurs cherchent donc à prouver la qualité de leurs solutions en développant des bornes inférieures de qualité assez bonne afin de les comparer avec leurs résultats. Néanmoins, pour avoir de bonnes bornes, il faut souvent un temps de calcul assez grand, ce qui reste une limite considérable.

Une attention particulière sera apportée aux travaux traitants :

- les problèmes d'équilibrage de lignes avec machines parallèles ;
- les problèmes d'équilibrage de lignes avec des temps inter-opératoires.

2.3.3. Problème d'équilibrage de lignes d'assemblage avec stations parallèles

Plusieurs travaux de recherche ont été consacrés à l'étude des problèmes d'équilibrage des lignes d'assemblage avec stations parallèles. En effet, le dédoublement des machines sur une même station de travail a plusieurs avantages (Buxey, 1974), tels que : la réduction des temps improductifs, le respect du temps de cycle (dans le cas où il y aurait des opérations qui ont un temps opératoire supérieur au temps de cycle), l'amélioration du taux de production imposé par l'opération la plus longue, la réduction des temps de transport, etc.

(Buxey, 1974) présente un cas d'étude du problème d'équilibrage des lignes d'assemblage avec stations parallèles. L'objectif principal du problème consiste à minimiser le temps mort total (temps improductif) de la ligne. L'auteur propose deux méthodes de résolution, la première (Optimal Station Program) consiste à utiliser une heuristique (Positional Weight Technique, PWT) qui est basée sur RPW (Helgeson et Birnie, 1961), la deuxième (Random Generation Program, RGP) est une instanciation de COMSOAL qui consiste à affecter les opérations aléatoirement aux stations. Les opérations affectées sont choisies à partir d'une liste d'opérations qui n'ont pas de contraintes de précédence et dont les temps opératoires ne dépassent pas le temps encore disponible sur la station courante. Des résultats numériques pour différentes valeurs du temps de cycle sont rapportés. L'auteur a conclu que RGP est beaucoup plus performant que PWT pour les problèmes de grande taille.

Pinto et al. (1975) traitent le problème d'équilibrage des lignes d'assemblage avec stations parallèles. Une formulation du problème en programme en nombres mixtes est présentée. Les auteurs proposent une procédure par séparation et évaluation. Un exemple numérique est présenté pour illustrer la méthode. Pinto et al., (1981) étudient le même problème mais avec

un double objectif, la diminution du coût de fonctionnement de la ligne et la maximisation de son taux de production.

Bard (1989) traite un autre problème d'équilibrage de ligne d'assemblage avec machines parallèles. L'objectif consiste à trouver un compromis entre le nombre de stations de travail et le coût d'installation de nouvelles stations. Un algorithme de programmation dynamique est développé. Cet algorithme est une version adaptée pour l'équilibrage de lignes sérielles tenant en compte du coût de la mise en parallèle des stations. Une borne inférieure est calculée pour simplifier la résolution. Des expériences numériques sur 15 problèmes différents (15, 25 et 40 opérations) sont présentées. Les auteurs montrent l'efficacité de leur méthode puisqu'elle permet la résolution de tous les problèmes avec un temps de calcul raisonnable.

Askin et Zhou (1997), proposent un programme non linéaire en nombre entier pour la résolution d'un problème d'équilibrage des lignes de production. L'objectif principal est de minimiser le coût d'acquisition et le coût de fonctionnement de la ligne. Le modèle étudié est multi-produit et il se caractérise par la possibilité de mettre des stations en parallèle à chaque niveau de la ligne. Le problème consiste à assigner les opérations aux stations correspondantes suivant le type de produit. Une méthode par construction est proposée, elle consiste à affecter les opérations selon leur poids moyen par rapport à l'ensemble des opérations à assigner (Weighted average task set). Des résultats numériques sont présentés, les instances étudiées variant selon le nombre d'opérations (30 ou 100), le rapport entre le coût de l'équipement et celui des outils (1 ou 30) et le paramètre de contrôle des temps d'exécution des opérations qui sont générés aléatoirement (p : moyenne de la loi de distribution aléatoire des temps opératoires, elle est égale à 2 ou 5). Ainsi huit cas d'étude sont analysés, les auteurs remarquant que l'architecture de la ligne est affectée par la variation de ces paramètres. La qualité des solutions est estimée avec une borne inférieure. La méthode permet donc de donner des solutions rapidement pour des problèmes de grande taille avec une estimation de leur qualité.

McMullen et Frazier (1998) traitent le problème d'équilibrage de lignes d'assemblage avec stations parallèles, avec des temps opératoires stochastiques. Ils développent une méthode de recherche par voisinage qui utilise le recuit simulé. Le problème est multi-objectif, l'objectif principal étant de minimiser le coût total de la ligne tout en cherchant à trouver une solution qui donne le temps de cycle le plus petit possible. Les auteurs effectuent des tests expérimentaux sur sept problèmes différents en utilisant 23 méthodes et approches déjà

développées dans la littérature, 6 d'entre elles sont basées sur le recuit simulé. Ils proposent ensuite une comparaison entre les différentes méthodes à l'aide de quatre critères de performance, à savoir : le coût de fonctionnement moyen, le nombre de pièces produites dans le temps de cycle et la moyenne du taux d'utilisation du système (par unité de temps). Ils remarquent que les méthodes utilisant le recuit simulé ont surtout un effet direct sur le temps de cycle (meilleur que celui des autres méthodes) et le coût total (moyenne qualité).

Dans (Süer, 1998), l'auteur propose une méthode de conception de lignes d'assemblage parallèles où chaque ligne peut contenir des stations parallèles. L'objectif principal consiste à minimiser le nombre d'opérateurs dans la ligne, tout en assurant un taux de production donné. La méthode se décompose en trois phases, la première consiste à regrouper les opérations en sous-ensembles suivant la configuration de la ligne, la seconde permet de déterminer pour chaque configuration le nombre de stations à mettre en parallèle, et la troisième, suivant le taux de production souhaité, donne le nombre de lignes à mettre en parallèle.

Gadidov et Wilhelm (2000) proposent une approche de résolution pour les problèmes de conception des systèmes d'assemblage. Le modèle étudié est une ligne d'assemblage mono-produit avec choix d'équipements. Deux cas sont traités, le premier avec des stations composées de machines parallèles qui effectuent les mêmes opérations sur des unités de produits différentes et le second avec des affectations imposées dues aux contraintes entre les opérations qui ne peuvent pas être effectuées sur le même poste. L'objectif est de minimiser le coût de la ligne (coût d'équipement + coût de fonctionnement). L'approche de résolution est basée sur une procédure de Branch and cut. L'approche comporte une heuristique (opération de moindre coût) pour avoir une borne supérieure, des pré-traitements pour diminuer le nombre des variables et deux méthodes de génération de coupes, une par la méthode FGP (Facet Generation Procedure) et l'autre par relaxation LP. Des expériences numériques sont présentées, 102 instances générées aléatoirement étant ainsi étudiées.

Vilarinho et Simaria (2002) proposent un nouveau modèle linéaire avec variables mixtes pour la programmation mathématique du problème d'équilibrage des lignes d'assemblage pour plusieurs produits avec stations parallèles et des contraintes de compatibilités entre les opérations. L'objectif est de minimiser le nombre total de machines. La procédure de résolution est composée de deux étapes, la première consiste à trouver une solution initiale en utilisant l'heuristique RPW, la seconde étape consiste à utiliser le recuit simulé pour améliorer la solution initiale. Une borne inférieure est déterminée pour permettre l'évaluation des

résultats. Des expériences numériques ont été présentées pour analyser l'efficacité de la méthode, il s'avère que pour les problèmes de petite taille (8 à 11 opérations et 2 ou 3 types de produits), la solution optimale est obtenue en un bon temps de calcul, tandis que les problèmes de grande taille (de 21 à 70 opérations et 2 ou 3 types de produits) nécessitent plus de temps (de 3 à 20 fois plus que les problèmes de petite taille) pour des résultats de qualité plus ou moins bonne (le meilleur a un écart 6,7% par rapport à la borne inférieure tandis que le plus mauvais est à 20% de sa borne).

Bukchin et Rubinovitz (2002) étudient le problème de conception des lignes d'assemblage flexibles avec choix d'équipements. L'objectif du modèle consiste à trouver une configuration qui engendre un coût d'investissement minimum, tout en prenant en compte le temps de cycle. Un algorithme optimal de séparation et évaluation est développé pour la résolution des problèmes de taille moyenne. L'efficacité de l'algorithme est améliorée grâce à l'utilisation d'une bonne borne inférieure, ainsi que des règles de dominance permettant de simplifier le modèle en diminuant le nombre de variables. En plus de la méthode exacte, un algorithme heuristique a été suggéré pour la résolution des problèmes de grande taille. La taille du problème est déterminée suivant : (i) le nombre d'opérations (entre 15 et 30), (ii) le nombre d'équipements alternatifs (entre 3 et 5), (iii) la variabilité de la durée d'exécution des opérations d'un équipement à l'autre, (iv) la flexibilité de la séquence d'assemblage et (v) la flexibilité des équipements d'assemblage. Des expériences numériques sont présentées et analysées. Trois critères de performance sont ainsi pris en compte : (i) le nombre total de nœuds de l'arbre d'évaluation, (ii) le nombre maximum de nœuds ouverts et (iii) le temps de calcul de l'algorithme. Ainsi, en variant les données du problème, les auteurs concluent que la méthode exacte est performante pour la résolution des problèmes de taille moyenne tandis que pour les problèmes de grande taille, il s'est avéré que la méthode heuristique est plus efficace.

2.3.4. Problème d'équilibrage de lignes d'assemblage avec temps de changement

Le problème d'équilibrage des lignes d'assemblage avec temps de changement (Sequence-dependent Setup-Time Assembly Line Balancing Problem, SDALBP) prend en compte les temps improductifs entre les opérations. Ces temps varient suivant la séquence dans laquelle les opérations sont exécutées. Par conséquent, pour calculer le temps de travail d'une station, les temps inter-opératoires doivent être considérés à chaque fois qu'une opération est affectée

après une autre sur la même station de travail. Le problème a été défini récemment et peu de travaux de recherche ont été consacré à son étude.

Dans Scholl et al. (2008), les auteurs définissent un problème de ce type, c'est-à-dire, SDALBP. Ils proposent plusieurs formalisations du problème : un programme mixte (MIP) et deux modèles nouveaux. L'objectif du problème consiste à minimiser le nombre total de stations dans la ligne. L'influence entre les opérations est due aux différentes interactions entre opérations. Deux types d'interactions sont définis : l'interaction unidirectionnelle (pour deux opérations *i* et *j*, seulement l'opération *j* influence l'exécution de l'opération *i*) et l'interaction bidirectionnelle (chacune des deux opérations (*i* ou *j*) restreint l'autre). Les auteurs étudient les connections entre le SDALBP ainsi formulé et SALBP. Ils montrent que le SALBP est une relaxation du SDALBP et que le SDALBP est une transformation du SALB (Figure 2.6). Ensuite, deux procédures d'optimisation sont présentées et détaillées. Des instances avec les différentes relations possibles entre les opérations sont générées. Des résultats expérimentaux sont reportés dans l'article. Les auteurs concluent que les deux approches développées proposent des solutions meilleures que celles trouvées par l'approche MIP.

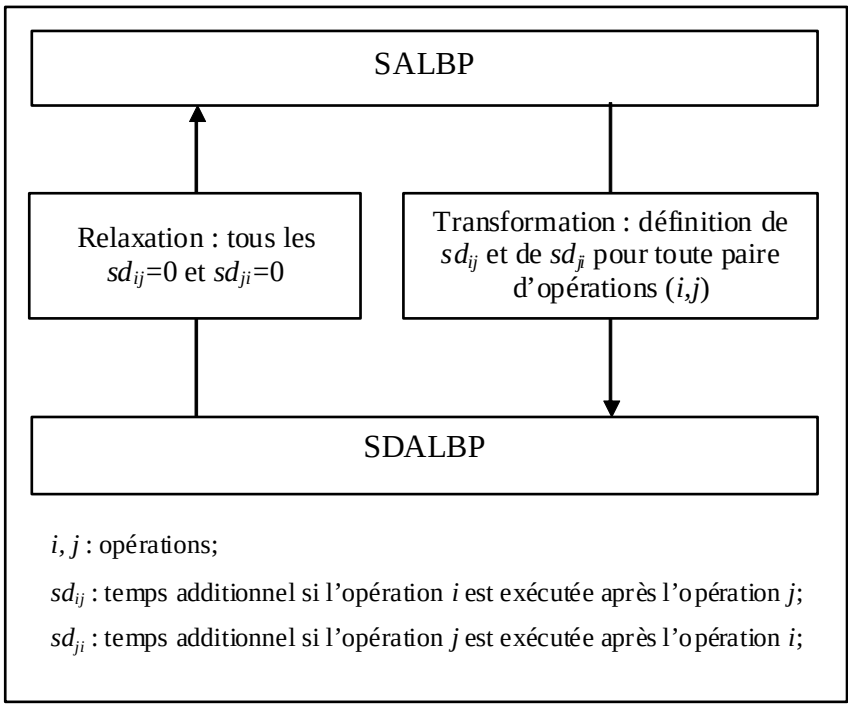

Figure 2.6 Liens entre le SDALBP et le SALBP

(Andrés et al., 2008) proposent un programme en nombres mixtes pour un problème avec des temps inter-opératoires. Le problème a été résolu à l'aide de Cplex. Les auteurs insistent sur

la difficulté d'obtenir une résolution exacte du problème dans un temps de calcul raisonnable. Des procédures heuristiques ont donc été développées. Ces procédures se basent sur deux stratégies heuristiques (Station oriented strategy heuristic et two operation oriented strategy heuristic) et quatre règles d'affectation. Ils utilisent la méthode GRASP. Une procédure pour le calcul d'une borne inférieure est également présentée. Des résultats numériques sont présentés et analysés. Les auteurs considèrent que les résultats obtenus sont encourageants mais ils envisagent comme perspectives d'appliquer d'autres métaheuristiques pour la résolution du problème.

Nous avons présenté des travaux traitants les problèmes d'équilibrage des lignes d'assemblage et en particulier, les problèmes avec des machines parallèles et tenant compte des temps inter-opératoires. Alors que les problèmes d'équilibrage de lignes d'assemblage avec machines parallèles ont été largement étudiés dans la littérature depuis la publication de Pinto et al. (1975), très peu de travaux de recherche ont été consacrés à l'étude des problèmes d'équilibrage des lignes d'assemblage avec des temps inter-opératoires dépendant de la séquence d'affectation. Dans notre étude, nous proposons un modèle dans lequel nous traitons à la fois les problèmes avec machines parallèles et les temps inter-opératoires en tenant compte des contraintes particulières des lignes de transfert dans l'usinage.

2.3.5. Problème d'équilibrage des lignes de transfert

Dans cette section, nous rapportons les principales méthodes utilisées ou développées pour la résolution des problèmes d'équilibrage des lignes de transfert. Les méthodes en question font partie des méthodes d'optimisation combinatoires.

Les problèmes d'équilibrage des lignes de transfert (TLBP) appartiennent aux problèmes d'équilibrage des lignes d'assemblage (GALBP). Ainsi, un TLBP possède les caractéristiques basiques du GALBP avec des caractéristiques et des contraintes sophistiquées. Les publications concernant le TLBP sont assez récentes. Il a été mentionné pour la première fois dans Szadkowski (1997). Dans Dolgui et al. (1999), le problème est défini pour les lignes de transfert dédiées. L'intérêt apporté au TLBP a été favorisé par les industriels du secteur de l'usinage qui cherchent eux aussi à minimiser les coûts afin d'améliorer leur compétitivité. La plupart des études concernant les TLBP se sont intéressées aux lignes de transfert dédiées équipées de machines à boîtiers multi-broches.

Deux principales catégories d'approches exactes pour la résolution des TLBP ont été utilisées dans la littérature, à savoir : les approches se basant sur les modèles MIP et l'utilisation des bibliothèques d'optimisation (comme Cplex) et les approches dédiées. Plusieurs méthodes d'optimisation ont été développées et mises en place. Les méthodes d'optimisation exacte pour le TLBP sont : (i) la programmation linéaire en nombres mixtes pour le dimensionnement de lignes d'usinage avec machines modulaires (Belmokhtar et al., 2006), (ii) la programmation dynamique (Dolgui et al., 1999) et la programmation linéaire en nombres mixtes (Dolgui et al., 2006b) pour l'équilibrage de lignes d'usinage avec des stations multi-broches, (iii) une procédure par séparation et évaluation pour la conception de lignes de transfert composées de stations avec une activation séquentielle des boitiers multi-broches (Dolgui et Ihnatsenka 2009).

Pour la résolution des problèmes de grande taille, plusieurs méthodes approchées ont été utilisées : (i) des heuristiques basées sur des règles de priorité (Guschinskaya et al., 2008 ; Finel et al., 2008) et (ii) des métaheuristiques, avec la méthode GRASP et les algorithmes génétiques (Dolgui et al,. 2008). Néanmoins, la plupart de ces travaux ont été consacrés à la résolution du problème d'équilibrage de lignes de transfert dédiées (stations à boitiers multi-broches, machines circulaires, etc.), et ils ne considèrent ni les temps inter-opératoires, ni les contraintes d'accessibilité, ni l'utilisation de machines parallèles.

2.4. Conclusion

Dans ce chapitre, nous avons présenté les principales méthodes de résolutions des problèmes d'optimisation combinatoire ainsi qu'un état de l'art des problèmes d'équilibrage des lignes de production. Nous avons mis l'accent sur les problèmes d'équilibrage de lignes avec stations parallèles, temps inter-opératoires, ainsi que l'équilibrage de lignes de transfert. Bien que plusieurs travaux académiques aient été consacrés à l'étude du problème d'équilibrage de lignes de transfert, aucun d'entre eux ne traite le problème que nous considérons dans ce mémoire. En plus, aucun problème d'équilibrage de lignes avec machines parallèles et temps inter-opératoires n'a été traité dans la littérature. Nous étudions donc un nouveau problème d'optimisation venant des applications industrielles dans le secteur de l'usinage.

Chapitre 3
Modélisation du problème d'équilibrage de lignes de transfert reconfigurables

Dans ce chapitre, nous nous intéressons au problème d'équilibrage de lignes de transfert avec centres d'usinage (CU) mono-broches. Ces lignes sont composées de stations de travail équipées de CU en parallèle. Les données et les contraintes qui caractérisent le problème sont détaillées et nous en proposons une modélisation linéaire en nombres mixtes. Ce modèle permet ainsi d'utiliser un solveur pour résoudre ce problème. En outre, une procédure de prétraitement spécifique est proposée pour faciliter la résolution en réduisant la taille du problème. Un exemple didactique est présenté, ainsi que des expérimentations numériques permettant de mesurer les limites de cette approche.

3.1. Définition du problème

Les lignes d'usinage étudiées sont équipées par des centres d'usinage (CU) à commandes numériques (CNC machine). En contraste avec les stations des lignes d'usinage dédiées (lignes d'usinage avec boîtiers multi-broches, Belmokhtar, 2006 ; Guschinskaya, 2007 ; voir Figure 3.1), les CU comportent une seule broche et un magasin d'outil (Figure 3.2). La pièce est fixée sur un plateau rotatif qui permet l'accès à ses différentes faces. Néanmoins, quelque soit le type du plateau rotatif utilisé (contraintes machines : axes de rotation) certaines faces restent inaccessibles. Ainsi, une contrainte d'accessibilité doit être prise en compte en fonction du plateau rotatif sur lequel la pièce est fixée (contrainte machine) et de la face d'appui de la pièce. Il faut également prendre en compte un temps additionnel entre deux opérations consécutives. En effet, dans une séquence d'opérations sur la même station, le passage d'une opération à une autre nécessite soit un changement d'outil, soit un déplacement d'outil et/ou une rotation de la pièce si les deux opérations ne concernent pas la même face.

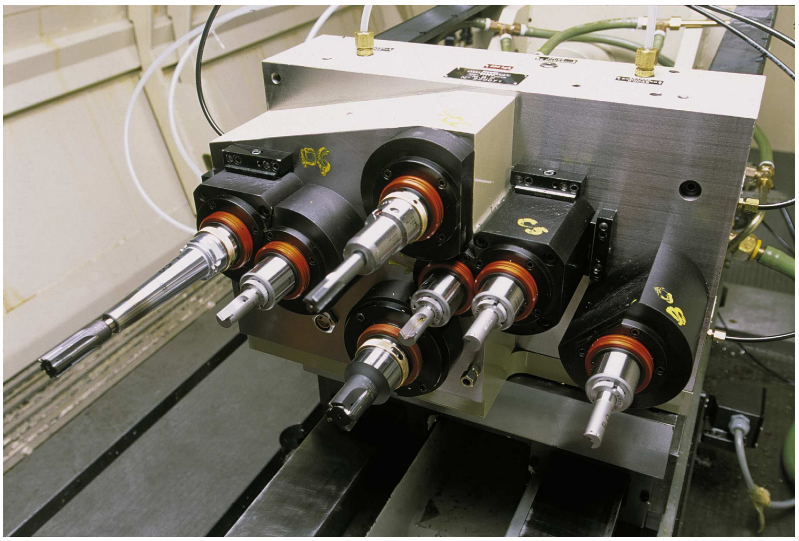

Figure 3.1 Boîtier d’usinage multi-broche (PCI/SCEMM)

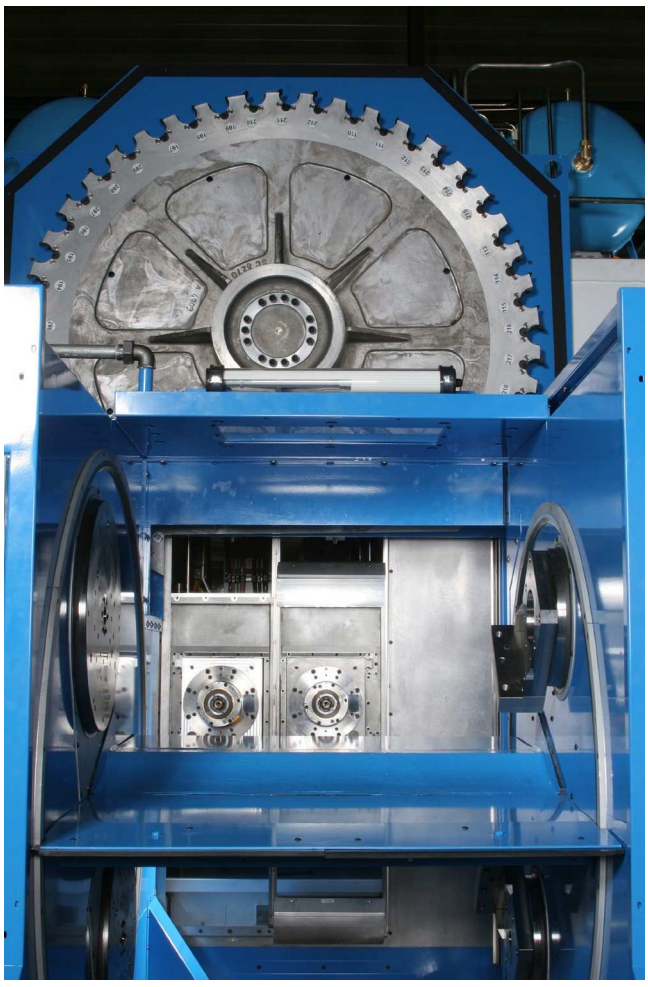

Figure 3.2 Centre d’usinage Meteor ML (PCI/SCEMM)

3.1.1. Données du problème

Le problème est caractérisé par un ensemble de données, à savoir : l'ensemble des opérations à affecter sur la ligne, les temps des opérations, les temps inter-opératoires entre les opérations (setup times) et les posages possibles (plateau rotatif et faces d'appui de la pièce). Dans la suite, nous utiliserons les notations suivantes :

- N : ensemble des opérations qui doivent être exécutées sur la ligne ;
- t_i : temps nécessaire pour l'exécution de l'opération i (pour toute opération i de N un temps opératoire fixe et connu est défini) ;
- t_{ij} : temps inter-opératoire pour le passage de l'opération i à l'opération j. La prise en compte de ce temps est nécessaire lors d'un changement d'outil (deux opérations successives nécessitant deux outils d'usinage différents), d'un déplacement d'outil (les deux opérations usinent la même face et utilisent le même outil d'usinage (Figure 3.3) et/ou lors de la rotation de la pièce si les deux opérations usinent deux faces différentes. Ce temps n'est pris en compte que si les deux opérations sont successives. Ainsi, le temps nécessaire à l'exécution d'un ensemble d'opérations sur la même station de travail dépend de la séquence dans laquelle ces opérations sont exécutées (Figure 3.4).

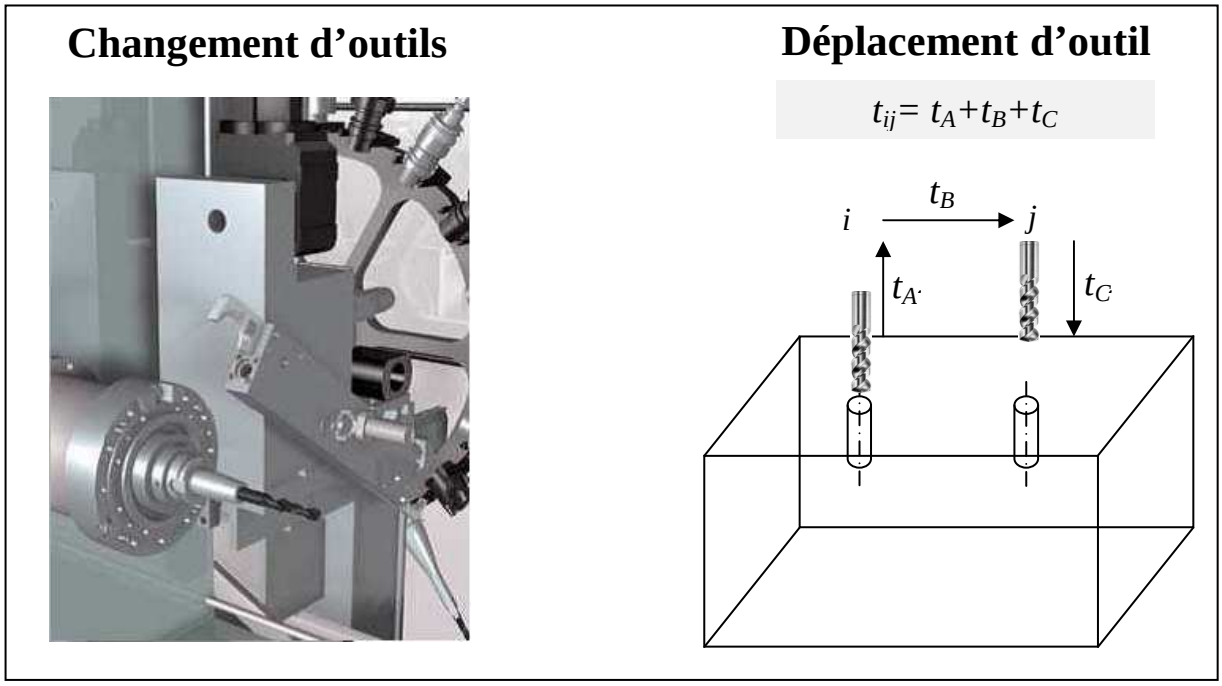

Figure 3.3 Centre d'usinage mono-broche : temps inter-opératoires

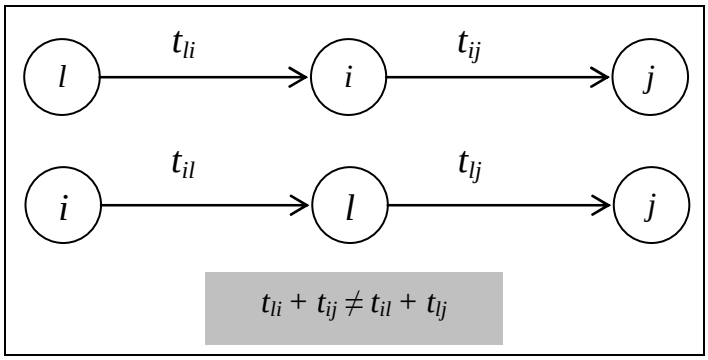

Figure 3.4 Temps inter-opératoires dépendant de la séquence

- *A* : ensemble des posages possibles sur les machines utilisées. Cet ensemble est déterminé à partir des caractéristiques mécaniques des machines ainsi qu'à partir des caractéristiques géométriques de la pièce à usiner. Dans la figure 3.5., la pièce est fixée sur un plateau rotatif qui peut avoir un ou deux axes de rotations (A et B). Si le plateau rotatif n'a qu'un seul axe de rotation (A), les deux faces latérales (hachurées en rouge) sont inaccessibles et bien sûr les opérations d'usinage les concernant ne peuvent pas être exécutées. Si le plateau possède deux axes de rotation (A et B), l'outil d'usinage peut accéder à toutes les faces latérales, mais la face d'appui n'est plus accessible (contraintes technologiques).

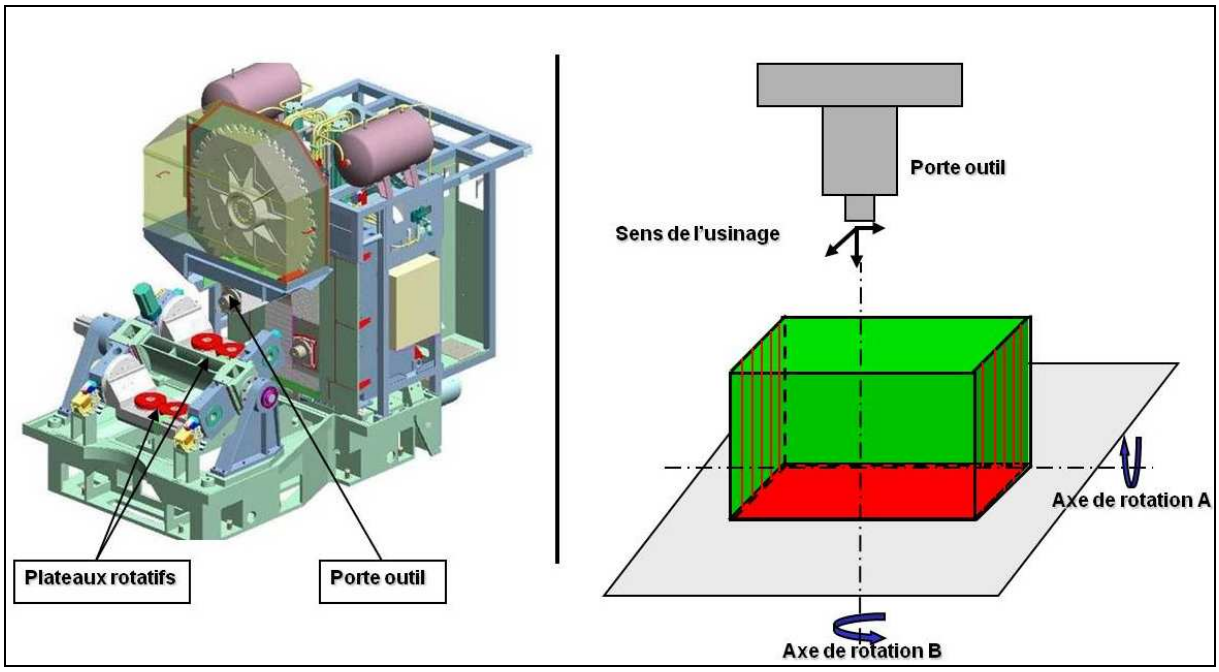

Figure 3.5 Axes de rotation d'une machine

3.1.2. Contraintes du problème

Comme nous l'avons déjà introduit, plusieurs contraintes doivent être prises en compte, notamment le temps de cycle, les contraintes de précédence entre les opérations, les contraintes d'inclusion et d'exclusion entre les opérations, les contraintes d'accessibilité (contraintes de posage) et le nombre maximum de CU par station. Toutes sont définies par les éléments suivants :

- T_0 : temps de cycle imposé. Le temps de cycle correspond au temps écoulé entre la production de deux pièces successives. T_0 correspond ainsi à une borne supérieure du temps de cycle. Le temps de cycle détermine ainsi la cadence maximale de la ligne (capacité de production). Dans une ligne synchrone, le temps de cycle est déterminé par la station de travail qui a le plus grand temps d'exécution auquel il faut ajouter le temps nécessaire pour déplacer la pièce d'une station à une autre. Afin que le temps de cycle soit respecté, il faut que le temps de cycle effectif de la ligne soit inférieur au temps de cycle imposé.

- P_i : ensemble des opérations prédécesseurs directs de i, c'est à dire qui doivent être exécutées avant que l'opération i commence. Ainsi, pour chaque opération i de N, toutes les opérations de P_i doivent être exécutées antérieurement à l'opération i. Plus exactement, pour toute opération $i \in N$, elle ne peut commencer son exécution avant que toutes les opérations de P_i terminent leur exécution. Ces relations de précédence peuvent être représentées par un graphe orienté $G_{or} = (N, D_{or})$ (où D_{or} est l'ensemble de couples d'opérations (i, j), tel que $i \in P_j$).

- ES : collection de relations d'inclusion entre les opérations. Chaque élément de ES est un sous-ensemble $e \subseteq N$ d'opérations qui doivent être exécutées sur la même station.

- $\overline{ES}$: collection de relations d'exclusion entre les opérations. Chaque élément de $\overline{ES}$ est un sous-ensemble $\overline{e} \subseteq N$ d'opérations qui ne peuvent pas être exécutées sur la même station.

- N_s : nombre maximum de stations de travail que peut comporter une ligne. Ceci est lié à l'agencement physique de la ligne et à l'espace disponible.

- n_0 : nombre maximal de CU par station de travail. Cette contrainte est imposée par des considérations d'espace et d'analyse de flux.

  Dans la Figure 3.6., nous illustrons une station de travail composée de trois centres d'usinage (en rouge). Ainsi, trois pièces différentes peuvent être affectées à la station (une pièce par CU). Chaque pièce est usinée pendant un temps de cycle propre à la station de travail (nombre de CU multiplié par le temps de cycle de la ligne, c.-à-d. $3*T_0$). Un régime spécial de chargement et de déchargement des pièces propre à la station est adopté afin d'assurer un flux continu de pièces entre les stations et éviter les encombrements.

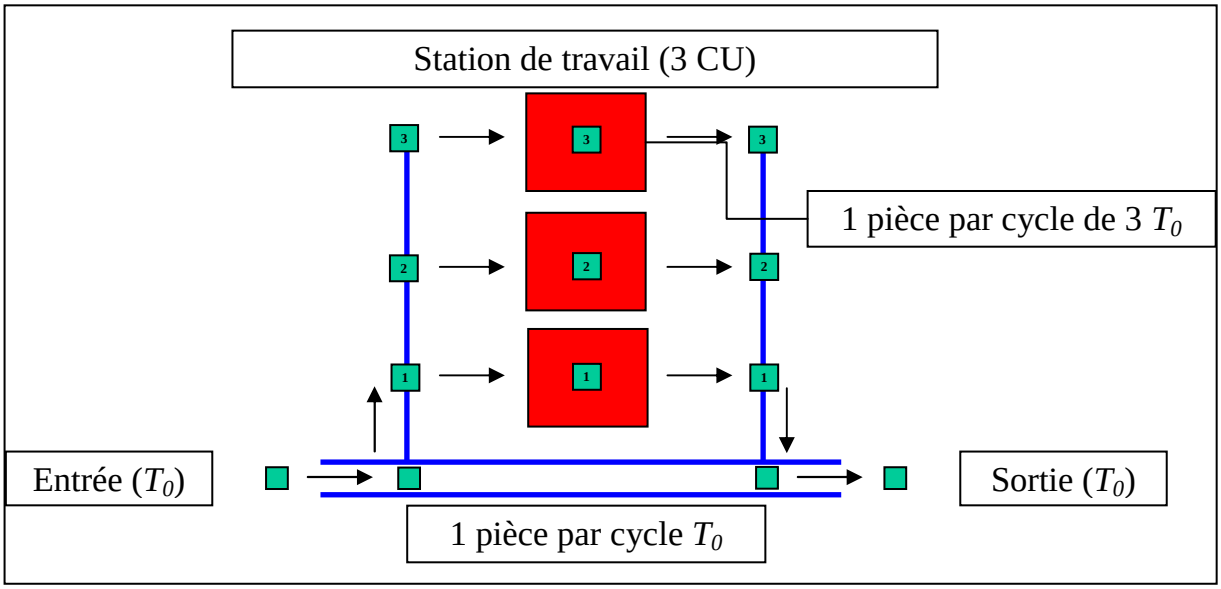

Figure 3.6 Station de travail à machines parallèles

- A_i : ensemble des posages possibles avec lesquels l'exécution de l'opération i est possible. Cette contrainte permet de spécifier les posages pour lesquels la face usinée par l'opération i est accessible (un posage est défini par la face d'appui de la pièce sur le plateau rotatif et les rotations permises par le type de machine utilisé).

3.2. Modélisation du problème

Nous commençons par introduire quelques notations supplémentaires.

Indices :

- i , j pour les opérations ;
- q pour l'ordre d'affectation d'une opération dans la séquence d'opérations ;
- k pour les stations de travail ;

- a pour le posage correspondant à la fixation de la pièce.

Paramètres:

- l_0 nombre maximum d'opérations autorisées à être affectées à une station de travail : chaque station de travail ne peut effectuer plus de l_0 opérations ;
- $q_0 = l_0 * Ns$: représente le nombre maximum de positions d'affectation possibles pour les opérations à affecter dans la séquence ;
- n_k pour le nombre de CU parallèles sur une station k ;
- P_i^* ensemble de tous les prédécesseurs de l'opération i (prédécesseurs directs et indirects) ;
- F_i ensemble des successeurs directs de l'opération i ;
- F_i^* ensemble de tous les successeurs de l'opération i (successeurs directs et indirects) ;
- LB_{ws} une borne inférieure pour le nombre de stations de travail ;
- $S(k)$ ensemble des positions possibles pour les opérations affectées à une station de travail k, cet ensemble est donné par un intervalle d'indices, l'intervalle maximum est : $S(k) = \{l_0 * (k-1)+1, l_0 * (k-1)+2, ..., l_0 * k\}$; $\forall k \in \{1, 2, ..., N_s\}$;
- $K(i)$ ensemble des stations de travail sur lesquelles une opération i peut être exécutée ; $K(i) \subseteq \{1, 2, ..., N_s\}$;
- $Q(i)$ ensemble des positions possibles pour une opération i dans la séquence à choisir de toutes les opérations, $Q(i) \subseteq \{1, 2, ..., l_0 * N_s\}$;
- $N(k)$ ensemble des opérations qui peuvent être exécutées sur une station de travail k ;
- $M(q)$ ensemble des opérations qui peuvent être affectées à la position q dans une séquence ;
- E_i station de travail au plus tôt à laquelle l'opération i peut être affectée ;
- L_i station de travail au plus tard à laquelle l'opération i peut être affectée.

Variables:

- $x_{iq} = 1$, si l'opération i est $q^{ème}$ (q est sa position dans la séquence d'affectation de toutes les opérations), sinon $x_{iq} = 0$;
- τ_q temps inter-opératoire nécessaire entre les opérations affectées sur la même station de travail aux positions q et $q+1$ (Figure 3.7) ;

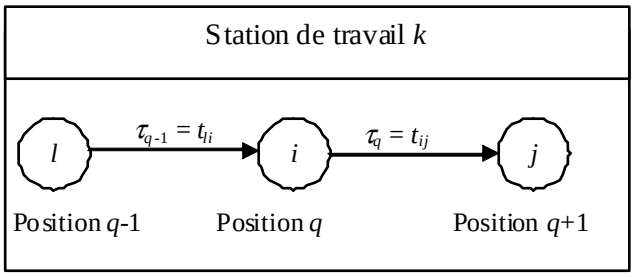

Figure 3.7 Définition de la variable τ_q

- $y_{nk} = 1$, si la station de travail k est composée de n CU fonctionnant en parallèle, $y_{nk} = 0$ sinon ;
- $z_{ka} = 1$ si on utilise le posage a sur la station de travail k, $z_{ka} = 0$ sinon.

Les contraintes du problème sont modélisées de la façon suivante :

- L'équation (3.1) indique qu'il faut choisir le nombre de CU pour chaque station. Il peut y avoir une seule variable supérieure ou égale à zéro (et aucune s'il n'y a pas de CU sur la station) :

$$\sum_{n=1}^{n_0} y_{nk} \le 1, \forall k = 1,2,\ldots,N_s \tag{3.1}$$

- L'équation (3.2) assure qu'une station de travail n'est ouverte que si la station précédente est également ouverte :

$$\sum_{n=1}^{n_0} y_{nk} \ge \sum_{n=1}^{n_0} y_{n(k+1)}, \forall k = LB_{ws}, LB_{ws}+1,\ldots,N_s-1 \tag{3.2}$$

- L'équation (3.3) assure que toute opération *i* est assignée une et une seule fois :

$$\sum_{q \in Q(i)} x_{iq} = 1, \forall i \in N \tag{3.3}$$

- La contrainte (3.4) assure qu'une position dans la séquence est occupée au plus par une opération :

$$\sum_{i \in M(q)} x_{iq} \leq 1, \forall q = 1,2,\ldots,q_0 \tag{3.4}$$

- L'équation (3.5) assure qu'une opération (sauf la première de chaque station) est affectée à une position seulement si une autre opération est affectée à la position précédente de la séquence (il n'y a pas de position vide entre deux opération sur une même station de travail) :

$$\sum_{i \in M(q)} x_{iq} \geq \sum_{p \in S(k);\, p>q} \sum_{i \in M(p)} x_{ip}, \forall q \in S(k) \setminus \max\{S(k)\}, \forall k = 1,2,\ldots,N_s \tag{3.5}$$

- L'équation (3.6) indique qu'au plus un posage est choisi pour chaque station de travail :

$$\sum_{a \in A} z_{ka} \leq 1, \forall k = 1,2,\ldots,N_s \tag{3.6}$$

- L'équation (3.7) assure que les contraintes d'accessibilité soient respectées :

$$\sum_{q \in S(k)} x_{iq} \leq \sum_{a \in A(i)} z_{ka}, \forall k \in K(i), \forall i \in N \tag{3.7}$$

- L'équation (3.8) calcule le temps additionnel entre une opération *i* et une opération *j* quand l'opération *j* est exécutée directement après l'opération *i* sur la même station de travail :

$$\tau_q \geq \sum_{j \in M(q+1) \setminus \{i\}} t_{ij} \cdot (x_{iq} + x_{j(q+1)} - 1), \;\; \forall i \in M(q), \forall q \in S(k) \setminus \max\{S(k)\}, \forall k = 1,2,\ldots,N_s \tag{3.8}$$

- L'équation (3.9) assure que le temps de traitement des différentes stations de travail n'excède pas le temps de cycle local qui correspond au nombre de CU installés sur la station multiplié par le temps de cycle objectif de la ligne :

$$\sum_{q \in S(k) \setminus \max\{S(k)\}} \tau_q + \sum_{i \in N(k)} \sum_{q \in S(k)} t_i \cdot x_{iq} \leq T_0 \cdot \sum_{n=1}^{n_0} n \cdot y_{nk}, \forall k = 1,2,\ldots,N_s \tag{3.9}$$

- L'équation (10) définit les contraintes de précédence entre les opérations :

$$1 + \sum_{q \in Q(j)} q \cdot x_{jq} \leq \sum_{q \in Q(i)} q \cdot x_{iq}, \forall i \in N, \forall j \in P_i \tag{3.10}$$

- L'équation (3.11) représente les contraintes d'inclusion :

$$\sum_{q \in S(k) \cap Q(i)} x_{iq} = \sum_{q \in S(k) \cap Q(j)} x_{jq}, \forall i, j \in e, \forall e \in ES, \forall k \in K(i) \cap K(j) \tag{3.11}$$

- L'équation (3.12) représente les contraintes d'exclusion :

$$\sum_{q \in S(k)} (x_{iq} + x_{jq}) \leq 1, \forall (i, j) \in \overline{ES}, \forall k \in K(i) \cap K(j) \quad (3.12)$$

- Les équations (3.13-3.16) donnent des contraintes additionnelles pour les valeurs possibles des variables :

$$0 \leq \tau_q \leq \max_{(i,j) \in N*N; i \neq j} (t_{ij}), \forall q = 1,2,\ldots,q_0 \quad (3.13)$$

$$x_{iq} \in \{0,1\}, \forall i \in N, \forall q \in Q(i) \quad (3.14)$$

$$y_{nk} \in \{0,1\}, \forall n = 1,2,\ldots,n_0, \forall k = 1,2,\ldots,N_s \quad (3.15)$$

$$z_{ka} \in \{0,1\}, \forall k = 1,2,\ldots,N_s, \forall a \in A \quad (3.16)$$

En fonction du cas traité, plusieurs critères sont proposés, nous considérons ainsi un des critères proposés, le modèle étant mono-objectif. Ceux-ci s'appuient essentiellement sur des combinaisons des éléments suivant :

- Nombre de machines ;
- Nombre de stations ;
- Équilibrage de la charge entre les stations de la ligne ;
- Écart entre la nouvelle et l'ancienne solution pour le cas d'une reconfiguration ;

Nous décrivons ci-dessous la manière d'exprimer dans notre modèle les différents critères.

- Le nombre de CU (3.17) dans la ligne correspond à la somme des machines sur l'ensemble des stations de travail :

$$\sum_{k=1}^{m_0} \sum_{n=1}^{n_0} n \cdot y_{nk} \quad (3.17)$$

Nous pouvons ainsi estimer le coût de la ligne en prenant en compte le coût d'un CU :

C_m : coût d'un CU ;

QC_m : la somme des coûts des CU installés dans la ligne ;

$$QC_m = \sum_{k=1}^{N_s} C_m . \sum_{n=1}^{n_0} n \cdot y_{nk} \quad (3.18)$$

Différents types de machines peuvent être utilisés sur la ligne. Nous considérons un coût machine selon son type qui est directement lié aux posages qu'elle autorise. Nous pouvons prendre en compte les types de machines installées sur la ligne, nous introduisons :

$C_m(a)$: coût d'une machine autorisant le posage $a \in A$;

Nous introduisons les équations (3.19-3.20) qui permettent de définir le coût d'une station de travail en fonction du type de machines installées et de leur nombre :

$$\rho_k \geq C_m(a) \cdot (\sum_{n=1}^{n_0} n \cdot y_{nk} - n_0 \cdot (1 - z_{ka})), \forall k = 1,..., N_s, \forall a \in A \quad (3.19)$$

$$\rho_k \geq 0, \forall k = 1,..., N_s \quad (3.20)$$

Le critère à minimiser est le suivant (3.21) :

$$\sum_{k=1}^{N_s} \rho_k \quad (3.21)$$

- Nombre de stations de travail (3.22) :

$$\sum_{k=1}^{m_0} \sum_{n=1}^{n_0} y_{nk} \quad (3.22)$$

Nous pouvons calculer le coût d'installation des stations comme suit (3.23) :

C_{ws} : Coût d'ouverture d'une station de travail ;

QC_{ws} : la somme des coûts de l'ouverture de toutes les stations de travail de la ligne ;

$$QC_{ws} = \sum_{k=1}^{N_s} C_{ws} . \sum_{n=1}^{n_0} y_{nk} \quad (3.23)$$

- Équilibrage de la charge (3.24) : il correspond à la somme des différences (en valeur absolue) entre la charge de chaque station et la charge moyenne de la ligne :

$$QS_{eq} = \sum_{k=1}^{N_{WS}} \left| t_{ut}(k) - \tilde{t}_{ut} \right| \quad (3.24)$$

Avec :

$t_{ut}(k)$: la charge de la station *k*;

$\tilde{t}_{ut}$: la charge moyenne par station de la ligne ;

N_{WS} : nombre de stations dans la ligne.

Pour linéariser l'équation (3.24), nous définissons deux variables supplémentaires :

μ_{nk} : taux d'utilisation de la station *k* contenant *n* CU ;

$w_m = 1$ si la ligne contient *m* station de travail, 0 sinon ;

Nous remplaçons l'équation (3.9) par les équations (3.25-3.26) pour le respect du temps de cycle :

$$\sum_{q\in S(k)\setminus\max\{S(k)\}}\tau_q+\sum_{i\in N(k)}\sum_{q\in S(k)}t_i\cdot x_{iq}=T_0\cdot\sum_{n=1}^{n_0}n\cdot\mu_{nk},\forall k=1,2,\ldots,N_s \quad (3.25)$$

$$\mu_{nk}\le y_{nk},\forall k\in\{1,2,\ldots,N_s\},\forall n\in\{1,\ldots,n_0\} \quad (3.26)$$

Les équations (3.27-3.28) permettent de calculer le nombre de stations sur la ligne :

$$\sum_{n=1}^{n_0}\sum_{k=1}^{N_s}y_{nk}=\sum_{m=1}^{N_s}m\cdot w_m \quad (3.27)$$

$$\sum_{m=1}^{N_s}w_m=1 \quad (3.28)$$

Les équations (3.29-3.30) permettent de calculer les écarts au taux d'utilisation moyen de la ligne :

$$\sum_{n=1}^{n_0}\mu_{nk}-\sum_{n=1}^{n_0}\sum_{k'=1}^{N_s}\frac{\mu_{nk'}}{m}\ge\delta_k^+-\delta_k^--N_s\cdot(1-w_m),\forall k,m\in\{1,\ldots,N_s\} \quad (3.29)$$

$$\sum_{n=1}^{n_0}\mu_{nk}-\sum_{n=1}^{n_0}\sum_{k'=1}^{N_s}\frac{\mu_{nk'}}{m}\le\delta_k^+-\delta_k^-+N_s\cdot(1-w_m),\forall k,m\in\{1,\ldots,N_s\} \quad (3.30)$$

Avec :

$0\le\mu_{nk}\le1,\forall k\in\{1,2,\ldots,N_s\},\forall n\in\{1,\ldots,n_0\}$;

$w_m\in\{0,1\},\forall m\in\{1,\ldots,N_s\}$;

$0\le\delta_k^+\le1,\forall k\in\{1,\ldots,N_s\}$;

$0\le\delta_k^-\le1,\forall k\in\{1,\ldots,N_s\}$;

Le critère d'équilibrage de la charge entre les stations est alors formulé de la façon suivante (3.31) :

$$QS_{eq}=\sum_{k=1}^{N_s}(\delta_k^++\delta_k^-)\ ; \quad (3.31)$$

- L'écart entre la nouvelle et l'ancienne solution pour le cas d'une reconfiguration :

Après l'implémentation de la ligne, il est possible que des changements de la demande interviennent (changement de gamme opératoire, changement au niveau du volume de production, changement de produit, etc.). Dans ce cas, une reconfiguration de la ligne est indispensable afin de répondre à la demande du client. Ainsi, un problème de reconfiguration de la ligne doit être défini. Ce problème consiste à affecter les opérations de la nouvelle

gamme opératoire aux stations déjà installées sur la ligne (et éventuellement ajouter de nouvelles stations). Dans ce cas, le critère à minimiser est la variation du nombre de CU par stations. Pour cela, nous considérons les modifications suivantes :

Nous ajoutons une équation additionnelle (3.32) et un nouvel ensemble de variables (3.33) qui permettent de formaliser l'écart entre le nombre de CU avant et après la reconfiguration pour chaque station de travail :

$$\sum_{n=1}^{n_0} n \cdot y_{nk} - \sum_{n=1}^{n_0} n \cdot y_{nk}^{Old} = \Delta_k^+ - \Delta_k^-, \forall k = 1,2,\ldots,N_s \tag{3.32}$$

$$\Delta_k^+, \Delta_k^- \geq 0, \forall k = 1,2,\ldots,N_s \tag{3.33}$$

Le nombre de machines par station de la solution initiale est récupéré grâce au paramètre (y_{nk}^{Old}). Le critère de reconfiguration à minimiser est défini comme suit (3.34) :

$$\sum_{k=1}^{N_s} \left(\Delta_k^+ + \Delta_k^-\right) \tag{3.34}$$

3.3. Prétraitement des données du problème

Les modèles linéaires proposés (équations (3.1–3.34)) peuvent être résolus en utilisant un solveur standard comme ILOG Cplex. Néanmoins, étant donné que ce problème est NP-difficile, le temps de calcul pourrait devenir prohibitif. En utilisant des techniques efficaces de prétraitement pour la réduction des variables (taille du modèle), le temps de calcul peut être diminué et ainsi la recherche de la solution optimale accélérée. Les ensembles ($K(i)$, $N(k)$, $S(k)$, $Q(i)$ et $M(q)$) et les stations au plus tôt et au plus tard de chaque opération sont initialisés de la façon suivante :

- $S(k) = \{0, ..., l_0 * N_s\}$; $\forall k \in \{1, 2, ..., N_s\}$;
- $K(i) = \{1, 2, ..., N_s\}, \forall i \in N$;
- $Q(i) = \{1, 2, ..., l_0 * N_s\}, i \in N$;
- $N(k) = N, \forall k \in \{1, 2, ..., N_s\}$;
- $M(q) = N, \forall q \in \{1, 2, ..., l_0 * N_s\}$;
- $E_i = 1, \forall i \in N$, la station au plus tôt est initialisée à la première station, pour toutes les opérations ;

- $L_i = N_s, \forall i \in N$ la station de travail au plus tard est initialisée à la dernière station possible, pour toutes les opérations.

Nous proposons une technique pour le calcul de bornes pour les indices possibles des variables des modèles mathématiques. Ceci peut simplifier le problème et réduire le temps de calcul. En tenant compte des différentes contraintes entre les opérations, nous pouvons calculer les ensembles $K(i)$, $N(k)$, $S(k)$, $Q(i)$ et $M(q)$ plus précisément. Nous notons que ces ensembles fournissent de possibles intervalles pour les valeurs des indices correspondants.

3.3.1. Procédure de prétraitement

Nous définissons les notations additionnelles suivantes :

- $E_i[r]$ est une variable récursive pour le calcul (étape par étape) de la valeur de E_i en tenant en compte des temps inter-opératoires entre les opérations, $r \in \{0,1\}$;
- $L_i[r]$ est une variable récursive pour le calcul (étape par étape) de la valeur de L_i en tenant en compte des temps inter-opératoires entre les opérations, $r \in \{0,1\}$;

En considérant les ensemble P_i^* (ensemble de tous les prédécesseurs de i) et F_i^* (ensemble de tous les successeurs de i), nous introduisons :

- $Sp_i[r]$: la somme des $(P_i^* - E_i[r]+1)$ plus courts temps inter-opératoires entre les opérations de $P_i^* \cup \{i\}$ (l'opération i et tous ses prédécesseurs, $i \in N$) ;
- $Sf_i[r]$: la somme des $(|F_i^*| - N_s + L_i[r])$ plus courts temps inter-opératoires entre les opérations de $F_i^* \cup \{i\}$ (l'opération i et tous ses successeurs, $i \in N$) ;
- $d[i, j]$: un paramètre de distance qui possède la propriété suivante: si $\{i, j\} \in \overline{ES}$, alors $d[i, j] = 1$, sinon $d[i, j] = 0$.

Le temps opératoire total T_{sum} (hors temps inter-opératoires) est calculé comme suit :

$$T_{sum} = \sum_{i \in N} t_i \ ;$$

Une borne inférieure du nombre des stations de travail peut être calculée en supposant que toutes ces stations sont composées de n_0 machines. Ainsi, le temps de cycle local des différentes stations de travail est égal à $(T_0 * n_0)$. La ligne devient une ligne sérielle composée

de stations de travail identiques avec le même temps de cycle local. La borne inférieure du nombre total de stations de travail peut être calculée comme suit :

$$LB_{ws} = \lceil T_{sum} / (T_0 \cdot n_0) \rceil$$

Notation: $\lceil x \rceil$ est le plus petit entier supérieur ou égal à x.

De la même façon, une borne inférieure du nombre de CU de la ligne (LB_m) peut être déterminée par l'expression suivante :

$$LB_m = \lceil T_{sum} / T_0 \rceil ;$$

La procédure proposée dans l'algorithme 3.1 permet de calculer les ensembles $K(i)$, $N(k)$, $S(k)$, $Q(i)$ et $M(q)$. Notons que les indices des opérations sont classés suivant l'ordre du graphe de précédence (dans un ordre topologique). Quelques lignes de l'algorithme sont commentées. Le symbole "//" est utilisé pour marquer le début et la fin des commentaires.

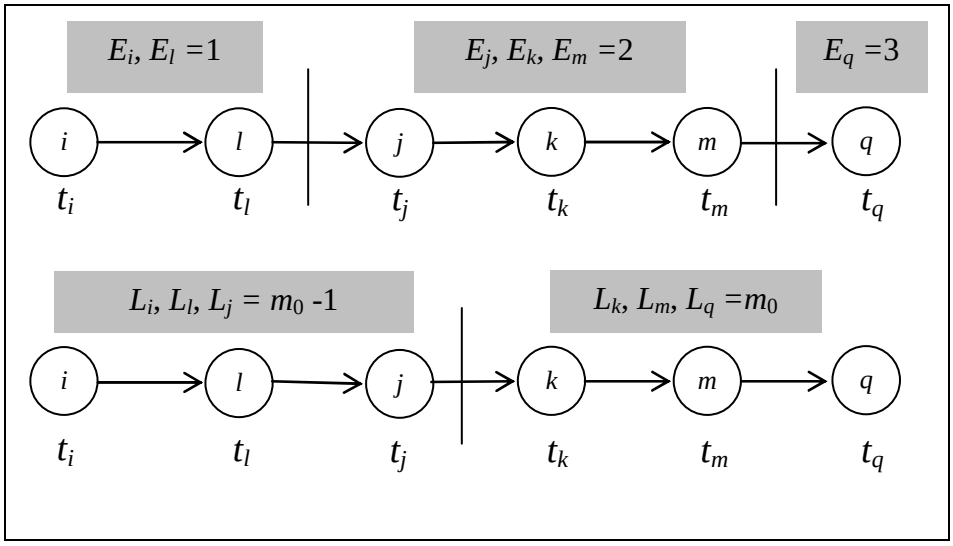

Figure 3.8 Calcul de E_i et de L_i en prenant en compte les contraintes de précédence

Dans les Figures 3.8 et 3.9, nous illustrons la méthode de calcul des valeurs des stations au plus tôt E_i (et des stations au plus tard L_i) pour toute opération $i \in N$. Pour chaque opération, nous savons qu'elle ne peut être exécutée que sur une station d'indice supérieur ou égal à celui de la station à laquelle est affecté l'un de ses prédécesseurs (resp. inférieur ou égal à celui de ses successeurs). Ceci nous permet de calculer, en utilisant les temps opératoires et inter-opératoires des prédécesseurs (resp. successeurs) ainsi que le temps de cycle, de définir l'indice de la station au plus tôt (resp. au plus tard) pour toutes les opérations.

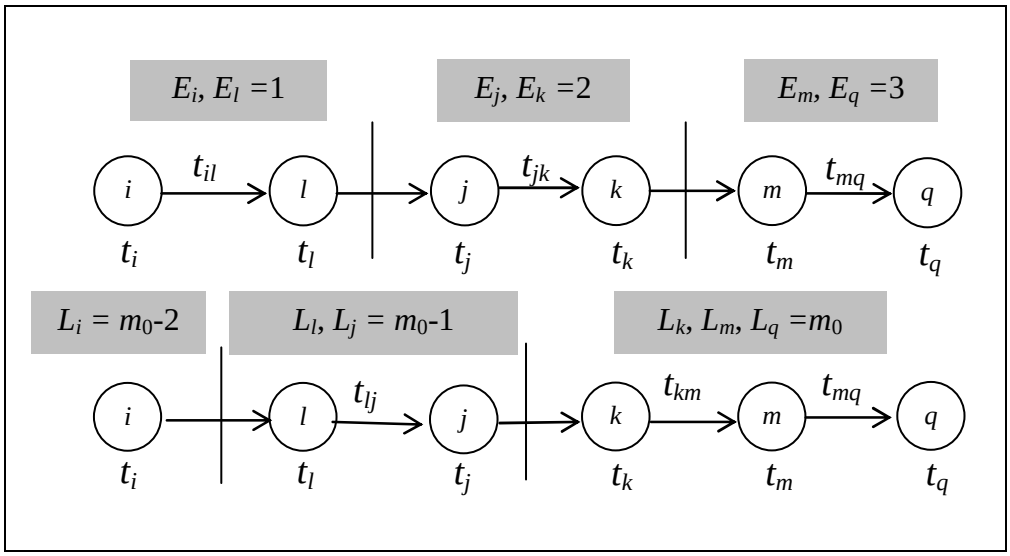

Figure 3.9 Calcul des valeurs de E_i et de L_i en tenant compte des temps inter-opératoires

Les contraintes d'exclusion peuvent être utilisées dans le calcul des indices E_i et L_i en les combinant avec les contraintes de précédence. Si une opération *i* est prédécesseur d'une opération *j*, et que ces deux opérations sont en exclusion, alors, elles ne peuvent pas être affectées à la même station et l'indice de station au plus tôt de *j* est strictement supérieur à celui de l'opération *i* (Figure 3.10).

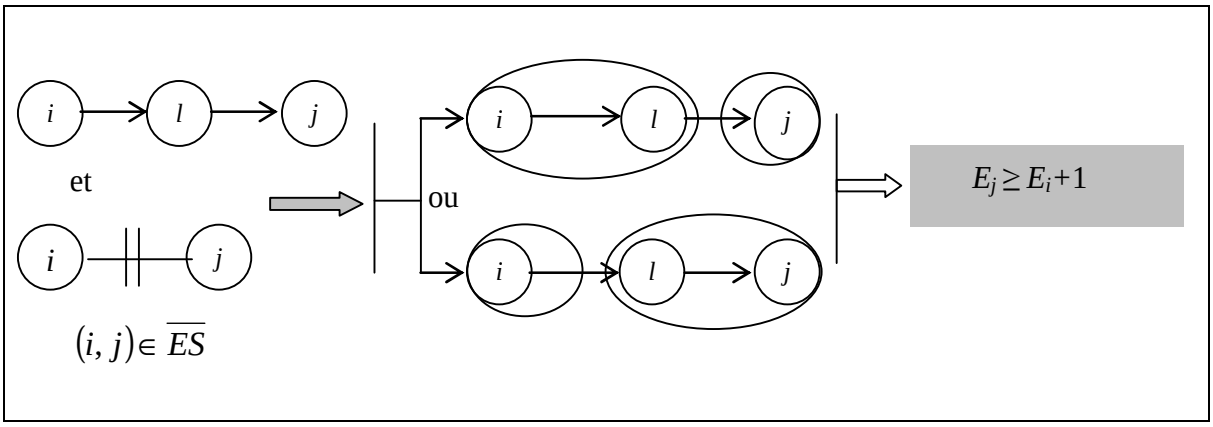

Figure 3.10 Calcul de E_i en tenant compte des contraintes d'exclusion

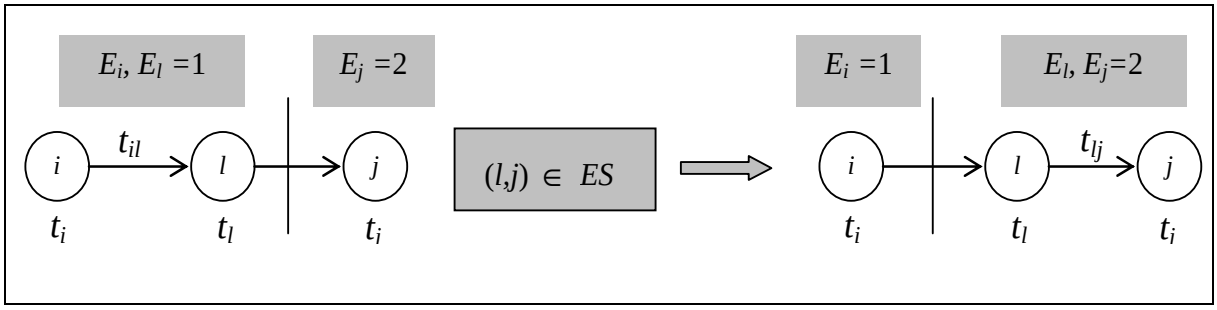

Figure 3.11 Calcul de E_i en tenant en compte les contraintes d'inclusion

Dans la Figure 3.11, nous illustrons l'utilisation des contraintes d'inclusion pour le calcul de l'indice de la station au plus tôt. Si deux opérations *l* et *j* sont en inclusion, alors, leur indice de station au plus tôt doivent être égaux. Ainsi, si $E_l \le E_j$ alors, $E_l = E_j$ et si $E_j \le E_l$ alors, $E_j = E_l$.

Nous présentons dans la suite le corps de l'algorithme de prétraitement.

***Algorithme* 3.1:**

Étape 1 // calcul de E_i et L_i en tenant compte des contraintes de précédence et des temps inter-opératoires //

Pour tout $i \in N$ **faire**

$E_i[0] \leftarrow \left\lceil (t_i + \sum_{j \in P_i^*} t_j)/(n_0 \cdot T_0) \right\rceil$ // calculer la station au plus tôt $E_i[0]$ sur laquelle l'opération i peut être exécutée en tenant compte des contraintes de précédence //

$L_i[0] \leftarrow N_s - \left\lceil (t_i + \sum_{j \in F_i^*} t_j)/(n_0 \cdot T_0) \right\rceil + 1$ // calculer la station au plus tard $L_i[0]$ sur laquelle l'opération i peut être exécutée en considérant les contraintes de précédence//

$E_i[1] \leftarrow \left\lceil (t_i + Sp_i[0] + \sum_{j \in P_i^*} t_j)/(n_0 \cdot T_0) \right\rceil$ // calculer $E_i[1]$ nouvelle valeur de E_i obtenue en tenant compte des temps inter-opératoires entre les opérations //

$L_i[1] \leftarrow N_s - \left\lceil (t_i + Sf_i[0] + \sum_{j \in F_i^*} t_j)/(n_0 \cdot T_0) \right\rceil + 1$ // calculer $L_i[1]$, c.-à-d. la nouvelle valeur de L_i obtenue par la prise en compte des temps inter-opératoires //

si $E_i[1] \neq E_i[0]$ **alors** $E_i \leftarrow \max\left(E_i[0]+1, \left\lceil (t_i + Sp_i[1] + \sum_{j \in P_i^*}(t_j))/(n_0 \cdot T_0) \right\rceil \right)$,

sinon $E_i \leftarrow E_i[1]$ // mise à jour des valeurs de E_i //

si $L_i[1] \neq L_i[0]$ **alors** $L_i \leftarrow \min\left(L_i[0]-1, N_s - \left\lceil (t_i + Sf_i[1] + \sum_{j \in F_i^*}(t_j))/(n_0 * T_0) \right\rceil + 1 \right)$,

sinon $L_i \leftarrow L_i[1]$ // mise à jour des valeurs de L_i //

Fin pour

Étape 2 // calculer E_i en tenant compte des contraintes d'exclusion et d'inclusion //

$j_{cur} \leftarrow 1$

Faire

$j_{\min} \leftarrow j_{cur}$; $j_{cur} \leftarrow |N|$

// Nouvelles valeurs de E_i calculées en considérant les contraintes d'exclusion //

pour $j \leftarrow j_{\min}+1,...,|N|$ **faire** $E_j \leftarrow \max\left(\max_{i\in P_j^*}\{E_i + d[i,j]\}, E_j\right)$ **fin pour**

pour tout $e \in ES$ **faire**

$E_e \leftarrow \max_{j\in e}(E_j)$

Pour tout $j \in e$ **faire: si** $E_j < E_e$ **alors**

$E_j \leftarrow E_e$ // E_i est calculé en tenant compte des contraintes d'inclusion //

$j_{cur} \leftarrow \min\{j_{cur}, j\}$

fin pour

fin pour

jusqu'à $j_{cur} = |N|$

Étape 3 // calculer L_i en tenant compte des contraintes d'inclusion et d'exclusion //

$j_{cur} \leftarrow |N|$

faire

$j_{\max} \leftarrow j_{cur}$

$j_{cur} \leftarrow 1$;

// Nouvelles valeurs de L_i calculées en considérant les contraintes d'exclusion //

pour $j \leftarrow j_{\max}-1,...,1$ **faire** $L_i \leftarrow \min\left\{\min_{i\in F_j^*}\{L_i - d[j,i]\}, L_j\right\}$ **fin pour**

pour tout $e \in ES$ **faire**

$L_e \leftarrow L_e \leftarrow \min_{j\in e}(L_j)$

Pour tout $j \in e$ **faire si** $L_j > L_e$ **alors**

// L_i est calculé en tenant compte des contraintes d'inclusion //

$L_j \leftarrow L_e$

$j_{cur} \leftarrow \max\{j_{cur}, j\}$

fin pour

fin pour

jusqu'à $j_{cur} = 1$

Étape 4 // calculer les ensembles $K(i)$, $N(k)$, $S(k)$, $Q(i)$ et $M(q)$ //

pour tout $i \in N$ **faire** $K(i) \leftarrow [E_i, L_i]$ **fin**

pour $k \leftarrow 1, 2, ..., N_s$ **faire**

$$N(k) \leftarrow \left\{ i \middle| \, i \in N, k \in K(i) \right\}$$

$$S(k) \leftarrow \left[1 + \sum_{k'=1}^{k-1} |S(k')|, \min\left(|N(k)|, l_0 \right) + \sum_{k'=1}^{k-1} |S(k')| \right]$$

fin pour

pour tout $i \in N$ **faire** $Q(i) \leftarrow \left[\min\{S(E_i)\}, \max\{S(L_i)\} \right]$ **fin pour**

pour $q \leftarrow 1, 2, ..., \max\{S(N_s)\}$ **faire** $M(q) \leftarrow M(q) \cup \left\{ i \middle| \, q \in Q(i) \right\}$ **fin pour**

Fin de l'algorithme

3.3.2. Illustration

Dans le but de mieux expliquer l'algorithme proposé, nous présentons un exemple numérique de 15 opérations. La figure 3.12. illustre le graphe de précédence ainsi que les temps opératoires.

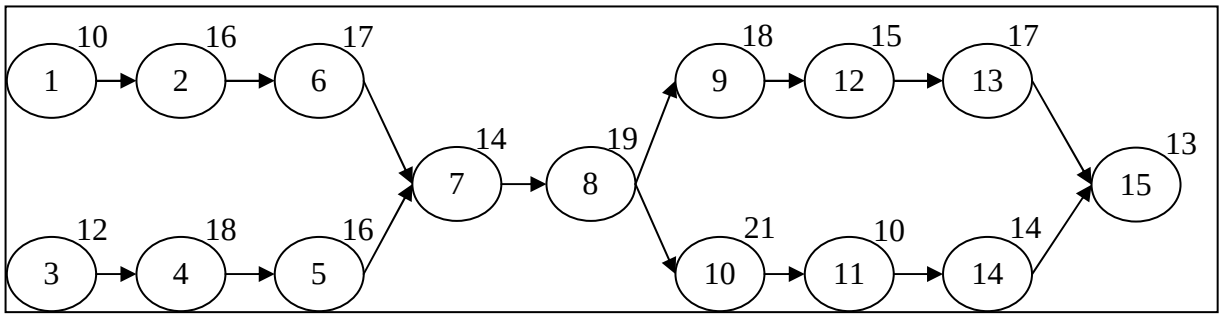

Figure 3.12 Graphe de précédence

Le temps de cycle objectif est : T_0=16 unités de temps; le nombre maximum de stations de travail : m_0=6; le nombre maximum de CU installables en parallèle sur une station : n_0=3; le nombre maximum d'opérations affectables sur une station : l_0=8.

Les contraintes d'inclusion sont: ES = {(2, 4); (6, 8); (5, 7), (11, 12); (14, 15)}.

Les contraintes d'exclusion sont : $\overline{ES}$ = {(2, 7); (1, 4); (5, 15); (6, 7); (8, 14)}.

Les temps inter-opératoires sont indiqués dans le Tableau 3.1. Par exemple, le temps $t_{4,5}$ = 3 correspond au temps de setup nécessaire quand l'opération 5 est exécutée immédiatement après l'opération 4.

Tableau 3.1 Temps inter-opératoires

*i**j*	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	-	4	4	3	2	1	1	2	1,5	1,5	1,6	3,6	2,8	2,4	1,2
2	5	-	1,5	1	1	2,5	3	3	3	3,5	4	4,3	4,9	4,9	3,5
3	1,5	3,5	-	2,5	1,2	3,4	4	4,2	3	2,2	4,3	3,8	2,9	1,8	3,9
4	4,5	4	3	-	3	1,5	3	2	4,5	4	4,9	4	2,3	1,6	1,8
5	4	4	5	5	-	2,5	2	2	4,5	1	1,6	4,4	4,2	3,3	1,7
6	25	1,5	3	1,2	1	-	2	2	3	4	4,3	1,1	3	3,6	2,6
7	4	3,8	3	1,8	4	3	-	4,7	4	1,4	2,5	1,7	4	4,3	3
8	3	2,5	4	3,4	3,2	2,4	1,6	-	2	4	2,2	4,7	2,1	2,3	1,5
9	3	4,9	4	2,3	1,6	3,6	3	1,2	-	3	1,4	2,8	4	3,7	3,1
10	1,5	4	1,5	4,6	3,7	2,2	2,7	1,2	4,8	-	1,7	2,3	1,6	4,9	2,8
11	4,7	1,1	4,5	2,1	1,8	4	2,6	1,8	3,4	3,4	-	2,8	3,3	3,5	4,4
12	1,7	3,9	3,2	4,6	1,9	1,7	3,4	3,7	3,3	2,4	2,9	-	3,9	3,5	1,4
13	4,6	3,4	3,8	1,6	3,6	4,8	2,2	3,2	2,2	1,6	1,4	4,4	-	3,9	1,6
14	3	2,7	2,6	2,1	2,8	3	1,6	2,2	4,2	4,8	4,4	3	1,2	-	3
15	3,4	4,8	1,5	4,4	3,6	2,2	1,4	3	1,8	2,1	4,6	3,2	3,6	1,4	-

Le temps total $T_{sum}= \sum_{i \in N} t_i$ = 161 unités de temps.

Une borne inférieure pour le nombre de stations peut être calculée comme suit : LB_{ws} = $\lceil T_{sum}/(T_0 * n_0) \rceil = \lceil 161/(16*3) \rceil = 4$

Ainsi, une solution ne peut pas avoir moins de 4 stations de travail.

Une borne inférieure pour le nombre de CU est : $LB_m = \lceil T_{sum} / T_0 \rceil = \lceil 161/16 \rceil = 11$.

Une solution ne peut pas avoir moins de 11 CU.

Nous appliquons la procédure de prétraitement sur notre exemple :

Étape 1: Les valeurs initiales de E_i et L_i pour les différentes opérations sont calculées en considérant les temps de setup et les contraintes de précédence (Figure 3.13):

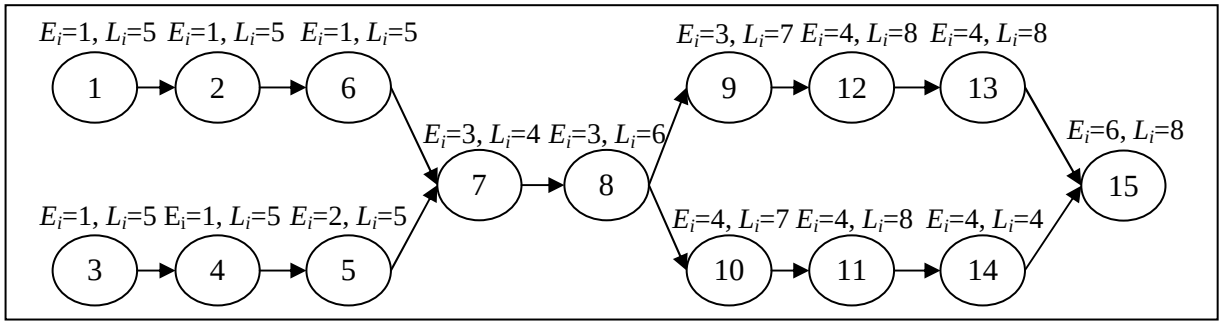

Figure 3.13 Valeurs de E_i et L_i obtenues en considérant les contraintes de précédence et des temps de setup

Étape 2: Les valeurs de E_i sont obtenues en tenant compte des contraintes d'inclusion et d'exclusion (Figure 3.14):

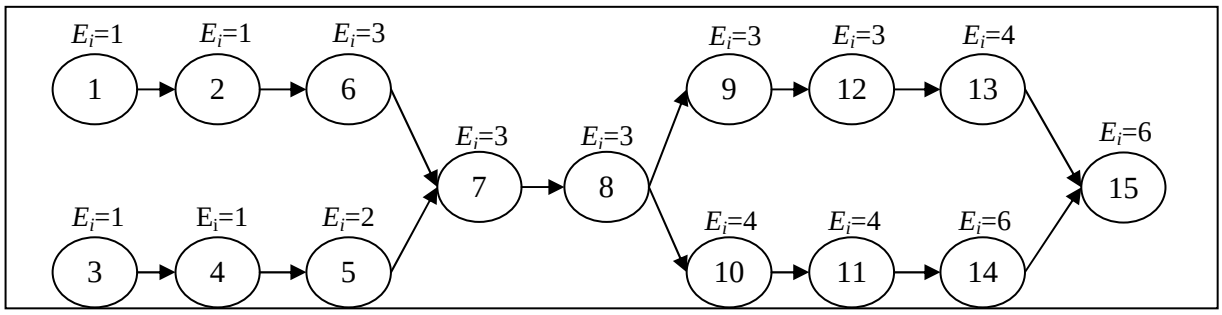

Figure 3.14 Valeurs de E_i en considérant les contraintes d'exclusion et d'inclusion

Étape 3: Les valeurs de L_i sont calculées en considérant les contraintes d'exclusion et d'inclusion (Figure 3.15):

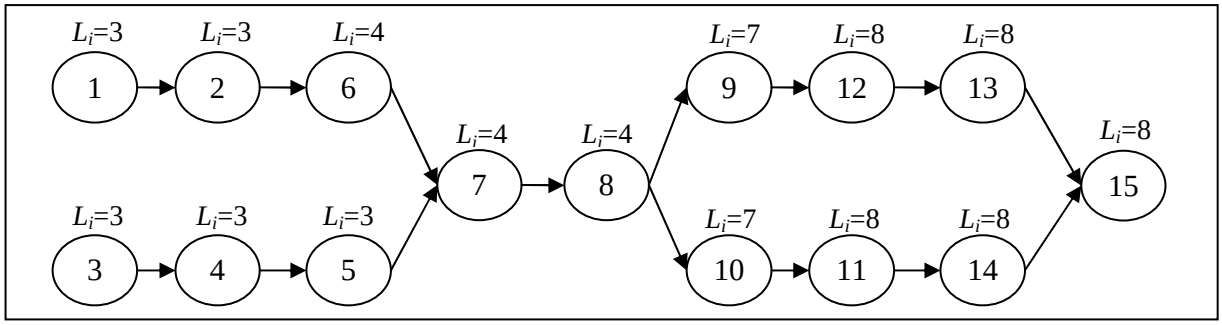

Figure 3.15 Valeurs de L_i en considérant les contraintes d'exclusion et d'inclusion

Étape 4: Les ensemble $K(i)$ pour les opérations $i \in N$ sont obtenus (Tableau 3.2).

Tableau 3.2 Les intervalles $K(i)$

Opération i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$K(i)$	[1,3]	[1,3]	[1,3]	[1,3]	[2,3]	[3,4]	[3,4]	[3,4]	[3,7]	[4,7]	[4,8]	[4,8]	[4,8]	[6,8]	[6,8]

Ensuite, les ensembles d'opérations $N(k)$ affectables aux stations de travail k=1, 2,…,m_0 sont définis (Tableau 3.3).

Tableau 3.3 Ensemble d'opérations $N(k)$ pour les stations de travail

St. k	1	2	3	4	5	6	7	8
$N(k)$	{1,2,3,4}	{1,2,3,4,5}	{1,2,3,4,5, 6,7,8,9}	{6,7,8,9,10, 11,12,13}	{9,10,11,12,13}	{9,10,11,12, 13,14,15}	{9,10,11,12, 13,14,15}	11,12,13, 14,15}

Les intervalles de positions $S(k)$ pour les opérations d'une station de travail k sont calculés, k=1, 2,…,m_0 (Tableau 3.4).

Tableau 3.4 Intervalles des positions $S(k)$ pour les stations de travail

St. k	1	2	3	4	5	6	7	8
$S(k)$	[1,4]	[5,9]	[10,17]	[18,25]	[26,30]	[31,37]	[38,44]	[45,49]

Finalement, l'intervalle des positions $Q(i)$ pour une opération i est donné, pour tout $i \in N$ (Tableau 3.5).

Tableau 3.5 Intervalles de positions $Q(i)$ pour les opérations

Op. i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$Q(i)$	[1,17]	[1,17]	[1,17]	[1,17]	[5,17]	[10,25]	[10,25]	[10,25]	[10,25]	[18,44]	[18,49]	[18,49]	[18,49]	[31,49]	[31,49]

L'application de cette procédure sur le modèle MIP (1)-(17) permet de diminuer le nombre de variables de 1096 à 758 et le nombre de contraintes de 1278 à 911.

3.4. Expérimentations

Nous avons testé le modèle (1)-(17). Afin d'évaluer les performances du modèle, nous avons généré aléatoirement plusieurs instances ayant les mêmes caractéristiques numériques que les cas industriels. Nous utilisons ILOG CPLEX 11.0 avec les paramètres par défaut pour la résolution de ces instances sur un ordinateur avec un processeur SUN UltraSPARC IIIi avec 1593 Mhz de CPU et 16 GB de mémoire. L'objectif des expérimentations consiste à : i) évaluer l'impact des prétraitements du modèle et ii) observer l'influence de deux caractéristiques du problème : le nombre d'opérations de l'ensemble N et la densité D_p du graphe de précédence. Pour cette dernière, nous nous appuyons sur la mesure introduite par Scholl (1999) et définie comme suit :

$$D_p = \frac{2 * \sum_{i \in N} \left| P_i^* \right|}{\left| N \right| * (\left| N \right| - 1)}$$

- $D_p = 0$, s'il n'y a aucune précédence entre les opérations ;
- $D_p = 1$, si le graphe de précédence est complet (c.-à-d., une seule séquence est possible).

Afin de générer des instances de différente complexité, nous avons considéré différentes tailles de problème (nombre d'opérations : 10, 12, 14, 16, 18 et 20) et différentes valeurs de la

densité (D_p) du graphe de précédence (10%, 25% et 40%). Les contraintes d'inclusion et d'exclusion sont générées avec des valeurs de densité fixes, respectivement 3% et 2%. Le temps de cycle T_0, le nombre maximum de machines par station de travail n_0 et le nombre maximum de stations de travail m_0 dans la ligne variant en fonction du nombre d'opérations de chaque instance considérée. Pour chaque jeu de données, nous générons aléatoirement 10 instances différentes.

L'objectif du modèle considéré consiste à minimiser le coût total de la ligne (éq. 3.18 + 3.23, coût machines + coût stations de travail). La qualité de l'algorithme est mesurée sur le temps de calcul (en seconde) qui correspond au temps nécessaire pour obtenir et prouver l'optimalité d'une solution, ou sur l'écart entre la meilleure solution trouvée et la borne inférieure quand ILOG Cplex n'arrive pas à résoudre le problème de façon optimale avec un temps de calcul limité (10 000 secondes).

Nous reprenons les résultats présentés dans Essafi et al. (2010a). Ils sont dans les Tableaux 3.6, 3.7 et 3.8. Premièrement, nous pouvons observer dans le Tableau 3.6 l'évolution du nombre moyen de variables du modèle en fonction du nombre d'opérations du problème (avant la phase de prétraitement). Puisque ce nombre ne dépend pas de la densité de précédence D_p, une seule valeur est rapportée pour chaque nombre d'opérations. Comme cela était prévisible, le nombre moyen de variables augmente avec le nombre d'opérations.

Tableau 3.6 Nombre moyen des variables

# Operations	10	12	14	16	18	20
# variables	560	773	1014	1288	1440	1753

Nous pouvons ainsi observer l'impact des procédures de prétraitement sur le nombre moyen de variables. Ces résultats sont rapportés dans le Tableau 3.7 dans lequel le taux de réduction (en pourcentage) obtenu est indiqué en fonction du nombre d'opérations et de la valeur de la densité D_p du graphe de précédence de l'instance.

Indépendamment du nombre d'opérations, l'impact des procédures de prétraitement est de plus en plus grand quand la densité du graphe de précédence augmente, sauf pour les instances de 20 opérations (D_p=25% / réduction=-8.65%, D_p=40% / réduction=-7.3%).

Notons que pour les cas avec des densités de précédence faibles (D_p =10%), les prétraitements n'ont qu'un effet limité sur le modèle initial (entre 3 et 5% de réduction).

Tableau 3.7 Réduction moyenne du nombre de variables après les prétraitements

D_p\#Op.	10	12	14	16	18	20
10%	-3,5%	-3,15%	-5,24%	-4,25%	-4,2%	-4,4%
25%	-9,7%	-9,13%	-10%	-7,14%	-7,11%	-8,65%
40%	-10,3%	-14,5%	-15,3%	-8,66%	-10,5%	-7,3%

Finalement, le Tableau 3.8 rapporte les résultats obtenus à l'aide de Cplex sur toutes les instances après avoir appliqué les prétraitements. Pour chaque instance, le temps de calcul moyen est indiqué, ainsi que l'écart entre la meilleure solution trouvée et la borne inférieure (cet écart est nul quand l'instance est résolue de manière optimale dans la limite des 10 000 secondes), et finalement le nombre d'instances résolues.

Tableau 3.8 Temps CPU moyen, écart moyen et nombre d'instances résolues avec Cplex

D_p\#Op.	10	12	14	16	18	20
10%	467,8s 0% 10	4 701s 4% 8	6 524,6s 9,17% 4	8 467,1s 12% 2	10 000s 19,2% 0	10 000s 19,6% 0
25%	181,9s 0% 10	2 173,2s 1,25% 9	5 931,5s 6% 5	7 433s 9% 3	9 730s 18% 1	10 000s 22% 0
40%	146,3s 0% 10	543,1s 0% 10	2 020,3s 1,5% 8	7 503,1s 11,8% 3	10 000s 16,9% 0	10 000s 21,2% 0

Nous remarquons que le temps de calcul augmente rapidement avec le nombre d'opérations. Aussi, la résolution est plus facile quand la densité de précédence est importante, ce qui est logique en considérant l'impact des procédures de prétraitement. Toutes les instances de taille 10, celles de taille 12 avec une densité de précédence égale à 40% ainsi que la plupart des autres instances de taille 12 (80% pour celles avec D_p=10% et 90% avec D_p=25%) sont résolues de façon optimale avec un temps de calcul raisonnable (<10 000 secondes). Pour plusieurs instances non résolues de façon optimale, l'écart correspond à un CU (écart=11,11%, 9,09%). Au delà de 16 opérations, peu d'instances ont été résolues de manière optimale et aucune avec 20 opérations.

3.5. Conclusion

Dans ce chapitre, nous avons présenté un nouveau problème d'équilibrage de lignes de transfert. Les lignes considérées sont équipées avec des centres d'usinage à commande numérique. Plusieurs centres d'usinage peuvent être installés en parallèles sur la même station de travail. Différentes contraintes doivent être considérées, à savoir : les contraintes de précédence, d'inclusion, d'exclusion et d'accessibilité. Le temps d'une station dépend de la séquence d'opérations. En effet, les centres d'usinage utilisés sont mono-broche, ce qui implique un temps supplémentaire pour le passage d'une opération à une autre.

Nous avons proposé une modélisation linéaire en nombres mixtes avec 5 critères différents ainsi qu'une procédure de prétraitement pour la réduction du nombre de variables. Les modèles ont été implémentés et testés et des résultats expérimentaux pour le modèle minimisant le coût de la ligne ont été rapporté. Ce modèle permet de résoudre de manière optimale certaines instances jusqu'à 18 opérations (spécialement avec une densité de précédence assez élevée) avec un temps de calcul acceptable. En effet, l'impact des prétraitements apparaît presque négligeable quand le graphe de précédence a une densité faible. Puisque les problèmes industriels peuvent comporter plus de 100 opérations, leur résolution ne semble pas possible avec l'approche proposée. Afin de résoudre des instances de plus grande taille, le développement d'approches heuristiques est nécessaire. Dans le chapitre suivant, nous développons différentes approches heuristiques et métaheuristiques.

Chapitre 4
Optimisation approchée des lignes de transfert reconfigurables

Dans le chapitre précédent, nous avons vu que les contraintes spécifiques aux lignes équipées de machines CNC rendaient la résolution exacte du problème beaucoup plus difficile que dans le cas d'autres lignes de transfert. Dans ce chapitre, nous proposons donc plusieurs approches heuristiques pour la résolution du problème d'optimisation des lignes de transfert reconfigurables. Les approches proposées sont toutes basées sur le principe des méthodes constructives qui ont prouvées leur efficacité pour la résolution de ce type de problème. D'abord, nous présentons un algorithme de construction gloutonne basé sur une règle de priorité. L'algorithme est testé et validé. Ensuite, quelques autres heuristiques gloutonnes sont développées en utilisant des règles de priorité et des techniques de type COMSOAL (Arcus, 1966). Puis, nous décrivons deux approches métaheuristiques, la première basée sur les colonies de fourmis, et la deuxième basée sur GRASP et intégrant une phase d'intensification de type Path-Relinking. Tous les algorithmes présentés sont testés sur un échantillon d'instances générées aléatoirement en respectant la structure des problèmes réels rencontrés dans l'industrie. Ceci nous permet d'évaluer et de comparer les performances des différentes approches développées.

4.1. Heuristiques gloutonnes et multi-départ

Dans cette section, nous proposons d'abord un algorithme glouton basé sur une approche constructive. La solution est obtenue de façon déterministe. Trois heuristiques basées sur cet algorithme sont proposées, nous avons fixé une règle de priorité pour chacune d'entre elles. Enfin, une heuristique de type COMSOAL avec un choix aléatoire en plusieurs itérations est mise en place.

4.1.1. Algorithme de construction gloutonne

L'algorithme proposé ici a été présenté dans Essafi et al. (2010b). Lors de la construction de la solution, nous devons respecter les contraintes et les données du problème. Pour cela, nous avons défini des règles pour construire une liste d'opérations candidates. Ensuite, une opération est choisie dans cette liste suivant une règle de sélection. Les listes sont ensuite mises à jour et le processus de sélection continue. La construction de la solution commence par la création d'une station comprenant un seul CU. Nous appelons « station courante » la station à laquelle nous affectons des opérations à l'étape courante de l'algorithme. Une fois qu'une station est créée, elle devient la station courante. Tant que nous pouvons ajouter des opérations à la station courante, nous ne créons pas de nouvelle station. Si la station courante est saturée et que nous pouvons encore ajouter des machines, nous ajoutons un CU de plus en parallèle et nous continuons l'affectation des opérations. L'algorithme ne prend fin qu'une fois toutes les opérations affectées. Dans l'Algorithme 4.1, nous présentons la structure générale de l'approche utilisée.

Afin de respecter les contraintes du problème, nous utilisons trois listes d'opérations :

- N_a : liste des opérations non encore affectées ;
- N_r : liste des opérations affectées ;
- N_c : liste des opérations candidates à l'affectation, c'est-à-dire qui :
  - n'ont pas de prédécesseurs non affectés ;
  - n'ont pas de contraintes d'exclusion avec les opérations déjà affectées à la station courante (directement ou indirectement par le biais d'une contrainte d'inclusion) ;
  - sont accessibles avec les posages possibles sur la station courante ;
  - ont un temps opératoire inférieur au temps de travail encore disponible sur la station courante.
- $LI(Op)$: liste des opérations devant être affectées à la même station que Op à l'étape courante de l'algorithme. Cette liste regroupe les opérations de N_a qui ont une contrainte d'inclusion avec Op, ainsi que les prédécesseurs directs et indirects de ces dernières (et les opérations en inclusion avec elles). Par exemple, si une opération i est sélectionnée, que i et j sont liées par une contrainte d'inclusion et que l est un prédécesseur de j non encore affecté, alors une liste $LI(i) = \{i, j, l\}$ est construite.

Nous ajoutons une liste complémentaire $\overline{N_c}$:

- $\overline{N_c}$: liste des opérations de N_c qui ne peuvent pas être affectées à la station courante (à cause du temps disponible sur la station qui est inférieur au temps opératoire additionnel qui sera engendré par l'ajout de n'importe quelle opération de $\overline{N_c}$, c'est à dire le temps de $LI(Op), \forall Op \in \overline{N_c}$).

Nous introduisons les nouvelles notations suivantes :

- $A_s(k)$: liste des posages possibles sur la station *k*. Afin de respecter les contraintes d'accessibilité de faces de la pièce, une liste des posages possibles est définie pour chaque station. Elle est initialisée avec l'ensemble de tous les posages. Ainsi, chaque fois qu'une opération est affectée, cette liste est mise à jour en supprimant les posages qui ne permettent pas l'exécution de cette opération.
- $T_s(k)$: temps opératoire disponible sur la station *k*. Il correspond à la différence entre le temps de cycle local de la station (nombre de CU * temps de cycle de la ligne) et la charge totale de la station (somme des temps opératoire+somme des séquences de temps inter-opératoires).
- $M(k)$: nombre de CU sur la station *k*.
- $N(k)$: séquence des opérations affectées à la station *k*. $N(k)(i)$ correspond à l'opération en $i^{ème}$ position dans l'ordre d'exécution sur la station *k*.

L'Algorithme 4.1 commence en créant une station comportant un seul CU. À chaque étape, nous commençons par construire la liste d'opérations N_c. Si N_c est vide, une nouvelle station est créée, devient la station courante (étapes 5, 6) et N_c est reconstruite. Pour toute opération *i* de N_c, nous construisons sa liste *LI*(*i*) (étape 9). Une fois les listes *LI* construites, nous sélectionnons une opération *Op* de N_c selon une règle de priorité et nous affectons toutes les opérations de la liste $LI(Op)$ (étape 11). Si le temps de travail disponible sur la station courante n'est pas suffisant, nous ajoutons un CU supplémentaire (étape 12). Si cet ajout est possible (étape 13), c.-à-d. ; le nombre de CU sur la station est strictement inférieur à n_0, les données de la station courante sont mises à jour (temps de travail disponible, nombre de CU, liste des opérations affectées, posages possibles, etc.), de même pour les listes N_a et N_c (étape 16). Si la condition n'est pas respectée, l'opération sélectionnée est mémorisée dans la liste $\overline{N_c}$ et supprimée de la liste N_c. (étape 4). Les opérations de *LI*(*Op*) sont supprimées de la

station courante (étape 17). L'algorithme s'arrête quand toutes les opérations sont affectées (N_a=Ø).

Algorithme 4.1 Heuristique gloutonne

Début

1: $k \leftarrow 1, N_a \leftarrow N, N_r \leftarrow \emptyset, \ \overline{N_c} \leftarrow \emptyset$

2: **Tant que** $(N_a \neq \emptyset)$ **Faire**

3: $$N_c \leftarrow \left\{ i \in N_a \left| \begin{array}{l} P_i^* \subseteq N_r \\ \quad et\,(t_i + t_{N(k)(i)}) \in [0, T_s(k)] \cup [T_0, (n_0 - M(k)) * T_0 + T_s(k)] \\ \quad et\, A(i) \cap A_s(k) \neq \emptyset \\ \quad et\, \forall j \in N(k), (i,j) \notin \overline{ES} \\ \quad et\, \forall j \in N(k), \overline{\exists} l \in N_a \left| (j,l) \in \overline{ES}, (i,l) \notin ES \right. \end{array} \right. \right\}$$

4: $N_c \leftarrow N_c \setminus \overline{N_c}$

5: **Si** $(N_c = \emptyset)$ **Alors**

6: $k \leftarrow k+1, \ \overline{N_c} \leftarrow \emptyset$

7: **Sinon**

8: **Pour** $(i = 1, ..., |N_c|)$ **Faire**

9: $LI \leftarrow \left\{ j \in N_a \left| \exists e \in ES, i \in e \, et \, j \in e \, ou \, \exists l \in e \right| j \in P_l^* \right\}$

10: **Fin Pour**

11: Sélectionner $i \in N_c \left| t_i = \max_{j \in N_c}(t_j) \right.$

12: $N(k) \leftarrow N(k) \cup LI(i)$

13: Mettre à jour $T_s(k), A_s(k)$

14: **Si** $(T_s(k) < 0)$ **Alors**

15: **Si** $(M(k) < n_0)$ **Alors**

16: $M(k) \leftarrow M(k)+1, \ N_a \leftarrow N_a \setminus LI(i), \ N_r \leftarrow N_r \cup LI(i),$ Mettre à jour $T_s(k)$

17: **Sinon** $N(k) \leftarrow N(k) \setminus LI(i), \ \overline{N_c} \leftarrow \overline{N_c} \cup \{i\}$

18: **Fin Si**

19: **Fin Si**

20: **Fin Si**

21 **Fin Tant que**

Fin

4.1.2. Règles de priorité

Comme nous l'avons déjà mentionné, pour chaque heuristique nous devons fixer une règle de priorité pour l'affectation des opérations candidates. Nous avons considéré les trois règles suivantes :

- TO (temps opératoire) : le critère de sélection consiste à sélectionner l'opération qui a le plus grand temps opératoire. Cela correspond à la règle proposée par Moodi et Young (1965) pour l'équilibrage de lignes d'assemblage ;
- TC (temps cumulé) : elle est définie à partir de la somme des temps opératoires de l'opération et de ses successeurs. Cela correspond à la règle RPW (Ranked Positional Weight) proposée par Helgeson et Birnie (1961). Nous la notons comme $pw(i)$ pour chaque opération i :

$$pw(i) = t_i + \sum_{j \in F_i^*} t_j \tag{4.1}$$

Ainsi, la règle de sélection est formulée de la façon suivante, l'opération qui est choisie est la plus grande vis-à-vis de ce critère.

- TCIP (temps cumulé avec prise en compte des inclusions et des posages) : ce critère permet de prendre en compte le temps opératoire des opérations des listes *LI*, les temps opératoires des successeurs directs des opérations de *LI* et le poids des opérations non encore affectées qui n'ont pas un posage commun avec les opérations de *LI*. Afin de calculer cette fonction, pour chaque opération i nous définissons un ensemble $N_{pos}(i)$ des opérations qui n'ont pas un posage commun avec les opérations de $LI(i)$:

$$N_{pos}(i) = \left\{ j \in N_a \,\middle|\, \exists l \in LI(i), A(j) \cap A(l) = \emptyset \right\}$$

La fonction de sélection gloutonne s'écrit de la façon suivante :

$$g_i = \left(\left(\sum_{j \in LI(i)} \left(t_j + \sum_{f \in F_j^*} t_f \right) \right) + \left(\frac{\left| N_{pos}(i) \right|}{\left| N_a \right|} * \left(T_0 - t_l(i) \right) \right) \right) \tag{4.2}$$

4.1.3. Approche itérative basée sur des choix aléatoires

Le schéma de base de cet algorithme est très proche de celui de COMSOAL (Computer Method of Sequencing Operations for Assembly Line) proposé par (Arcus, 1966) pour l'équilibrage des lignes d'assemblage (SALBP-1). Cette approche est basée sur un algorithme de construction avec sélection aléatoire de candidats. L'approche est itérative et une construction est lancée à chaque itération ce qui permet de générer un grand nombre de solutions. Ainsi, nous avons utilisé le principe de cette procédure en lançant l'Algorithme 4.1 un grand nombre de fois. La sélection des opérations est dans ce cas aléatoire. Nous gardons la meilleure solution qui correspond à la solution la moins coûteuse.

4.2. Colonies de fourmis

Les algorithmes de colonies de fourmis (Ant Colony Optimization, ACO) s'inspirent du comportement des fourmis lors de la recherche d'un chemin entre leur colonie et une source de nourriture. La méthode ACO a été proposée par Dorigo (1992) et a été formulée comme nouvelle métaheuristique dans Dorigo et Di Caro (1999). Elle a d'abord été appliquée pour la résolution du problème de voyageur de commerce (Dorigo et Gambardella, 1997), mais elle a aussi été appliquée sur un large éventail de problèmes d'optimisation combinatoire et notamment les problèmes d'ordonnancement et d'équilibrage de lignes (voir par exemple, Spicer et al. 2002 ; Bautista et Pereira, 2007 ; Sabuncuoglu et al. 2009 ; Berrichi et Yalaoui, 2010). Les algorithmes ACO correspondent à une procédure de construction itérative, les solutions étant construites élément par élément.

Les éléments d'une solution sont sélectionnés selon une probabilité basée sur l'évaluation heuristique g_i (Equation 4.2). L'algorithme proposé ici a été présenté dans Essafi et al. (2010c). Chaque fourmi construit une solution en suivant une probabilité lors de la sélection. Elle dépose ses phéromones selon la qualité de la solution générée. La mémoire collective des fourmis est utilisée pour faire apparaître les meilleures solutions (la piste de phéromones laissée par les fourmis est plus significative quand la solution est de meilleure qualité par rapport aux autres). Un élément de la solution correspond à l'affectation d'une opération à une station. À chaque itération, nous calculons la probabilité d'affectation pour chaque combinaison possible (opération/station) selon la qualité des solutions générées. Cette procédure est répétée un grand nombre de fois (jusqu'à atteindre un critère d'arrêt). Suivant l'avancement de la procédure, les éléments qui correspondent aux meilleures solutions possèdent des pistes de phéromones plus importantes que les autres, ce qui permet à l'algorithme de converger vers les solutions de meilleure qualité.

Pour la construction des solutions, nous utilisons l'algorithme de base présenté dans la section 4.1. Pour les paramètres de la colonie de fourmis nous introduisons les notations suivantes :

- m: le nombre total des fourmis ;
- τ_{ik} : la quantité de phéromones correspondant à l'affectation d'une opération i à une station k. Ce paramètre est mis à jour suivant la formule suivante :

$\tau_{ik} \leftarrow \rho * \tau_{ik} + \Delta\tau_{ik}$;

où ρ est le paramètre de conservation de mémoire, $(1-\rho)$ correspondant au facteur d'évaporation des phéromones ;

- Q : coût moyen des solutions générées par l'algorithme;
- QS_{Ant} : coût de la solution générée par la fourmi *Ant*.
- $\forall i,k,\ \Delta\tau_{ik} \leftarrow \sum_{Ant=1 \mid i \in N(k)}^{m} \frac{Q}{QS_{Ant}}$: la quantité de phéromone déposée à chaque itération par la fourmi *Ant*;

Pour chaque élément, nous définissons une probabilité suivant l'expression (4.3) :

$$\Pr_{ik} = \frac{[g_i]^{\alpha} * [\tau_{ik}]^{\beta}}{\sum_{j \in N_c} \left([g_j]^{\alpha} * [\tau_{jk}]^{\beta} \right)} \ ;\ \forall i \in N, k \in \{1, ..., N_s\} \qquad (4.3)$$

- α : paramètre d'importance de la fonction gloutonne ;
- β : paramètre d'importance de la piste de phéromones.

La sélection des opérations au sein de l'algorithme sera effectuée en utilisant une roulette basée sur la fonction de probabilité (Équation 4.3). Pour chaque fourmi, l'algorithme démarre, crée des stations et s'arrête lorsque toutes les opérations sont affectées. La structure générale de l'algorithme est présenté ci-après (Algorithme 4.2).

Nous lançons l'algorithme pour chaque fourmi (étape 3). Nous utilisons les mêmes listes de candidats que dans l'Algorithme 4.1. Avant la sélection d'une opération candidate, les valeurs des probabilités $\Pr_{ik}$ sont calculées pour toutes les opérations appartenant à la liste N_c (étape 12). Une fois la solution construite, les paramètres utilisés pour le calcul des probabilités sont mis à jour en fonction de la qualité de la solution générée (étape 29). Si le coût de la solution générée est inférieur au coût de la meilleure solution, cette dernière est remplacée (étape 28).

Algorithme 4.2 Colonie de fourmis

Début

1: $t \leftarrow 0; \tau_{ik} \leftarrow c, \forall i,k;\ coût(s^*) \leftarrow \infty,\ \overline{N_c} \leftarrow \emptyset$
2: **Tant que** $(t < condition\ d'arret)$ **Faire**
3: **Pour** $(Ant = 1,...,m)$ **Faire**
4: $k \leftarrow 1, N_a \leftarrow N$
5: **Tant que** $(N_a \neq \emptyset)$ **Faire**
6: Construire N_c
7: **Si** $(N_c = \emptyset)$ **Alors** $k \leftarrow k+1,\ \overline{N_c} \leftarrow \emptyset$
9: **Sinon**
10: **Pour** $(i = 1,...,|N_c|)$ **Faire**
11: Construire $LI(i)$, Calculer g_i (Equation 4.2) et $\Pr_{ik}$
13: **Fin Pour**
14: Sélectionner un élément $i \in N_c$ en utilisant $\Pr_{ik}$
15: Mettre à jour $N(k)$
16: **Si** $(T_s(k) < 0)$ **Alors**
17: **Si** $(M(k) < n_0)$ **Alors**
18: $M(k) \leftarrow M(k)+1$, Mettre à jour N_a, N_r, $T_s(k)$
20: **Sinon** $N(k) \leftarrow N(k) \setminus LI(i)$, $\overline{N_c} \leftarrow \overline{N_c} \cup \{i\}$
21: **Fin Si**
22: **Fin Si**
23: **Fin Si**
24: **Fin Tant que**
25: Calculer $coût(s)$
26: **Si** $(coût(s) < coût(s^*))$ **Alors**
27: $s^* \leftarrow s$
28: **Fin Si**
29: Mettre à jour $\tau_{ik}\ et\ QS_{Ant}$
30: **Fin Pour**
31: **Fin Tant que**

Fin

4.3. Heuristique basée sur GRASP

Dans cette section, nous proposons une approche d'optimisation basée sur la méthode GRASP (Greedy Randomized Adaptive Search Procedure) combinée à une phase d'intensification utilisant le Path-Relinking. GRASP est une métaheuristique développée par Feo et Resende (1989) qui a été appliquée avec succès pour la résolution de nombreux problèmes d'optimisation (voir Resende et al., 2006) et notamment pour des problèmes d'équilibrage de lignes d'assemblage (Andrés et al., 2008 ; Guschinskaya et al., 2008). Un

algorithme GRASP basique est une procédure de recherche locale multi-départ. Il est composé de deux phases : une phase de construction de solutions, et une phase d'amélioration.

La première phase consiste à générer des solutions admissibles à l'aide d'un algorithme glouton-aléatoire. La structure de l'algorithme proposé est basée sur l'Algorithme 4.1 présenté dans la section 4.1. Un paramètre de sélection aléatoire est utilisé afin de générer une famille de solutions différentes qui nous permet d'explorer mieux l'espace de solutions. La deuxième phase est une procédure de recherche locale qui permet d'améliorer la qualité des solutions obtenues en étudiant leur voisinage.

En plus du schéma de base de GRASP, nous avons ajouté une procédure d'auto-adaptation basée sur les idées de Reactive GRASP (Delmaire et al., 1999 ; Prais et Ribeiro, 2000). Cette procédure ajuste le paramètre aléatoire de l'algorithme en fonction de la qualité des solutions générées. La deuxième amélioration de l'algorithme consiste en une phase d'intensification basée sur le Path-Relinking. La procédure du Path Relinking a été initialement proposée par Glover (1996) comme une stratégie d'intensification explorant les trajectoires qui connectent les meilleures solutions obtenues par la recherche tabou ou scatter search (Glover et al., 2000). Laguna et Marti (1999) ont été les premiers à combiner un algorithme GRASP à du Path-Relinking. Cette combinaison a été utilisée avec succès pour l'optimisation de plusieurs problèmes complexes d'optimisation (voir par exemple, Delorme et al., 2004 ; Boudia et al., 2007 ; Alvarez-Valdes et al., 2008 ; Resende et al., 2010).

L'approche générale d'optimisation est résumée dans la Figure 4.1 Les lignes continues désignent les flux de solutions et les lignes discontinues désignent une prise de décision ou une mise à jour des paramètres. Cette procédure est itérative, elle est répétée jusqu'à ce qu'une condition d'arrêt soit vérifiée (nombre d'itérations, temps de calcul, qualité de solution, niveau d'amélioration par itération, etc.). Les meilleures solutions sont stockées dans une archive au fur et à mesure de l'avancement des itérations. Avant l'archivage des solutions, nous utilisons un modèle linéaire en nombres mixtes (MIP) pour optimiser les séquences d'opérations dans les stations. Le Path-Relinking est appliqué de façon périodique (après un nombre fixe d'itérations) sur les meilleures solutions de l'archive et, à la fin des itérations, sur toutes les solutions archivées. Les différentes composantes de l'approche proposée sont détaillées dans la suite.

4.3.1. GRASP

Comme nous l'avons déjà introduit, GRASP est une procédure multi-départ composée de deux phases : une phase de construction et une phase de recherche par voisinage. L'algorithme de construction est semi-glouton. Nous utilisons la fonction gloutonne proposée dans la section 4.1. avec un paramètre aléatoire pour la sélection des éléments à ajouter. La procédure de recherche locale est basée sur plusieurs voisinages à explorer ce qui nous permet d'obtenir un optimum local après chaque itération.

4.3.1.1. Phase de construction

La phase de construction est basée sur un algorithme semi-glouton. Cet algorithme à la même structure que celui proposé dans la première section de ce chapitre. Nous utilisons ainsi les mêmes listes de candidats. Comme l'Algorithme 4.1, l'algorithme semi-glouton commence par créer une station, et finit après l'affectation de toutes les opérations. Après la construction de la liste N_c et des ensembles $LI(i)$ pour les opérations i de N_c, nous calculons les valeurs de la fonction gloutonne g_i de toutes les opérations de N_c et récupérons les valeurs maximales $g_{\max}$ et minimales $g_{\min}$ de cette fonction (étape 10).

Nous définissons une liste restreinte de candidats $RN_c \subseteq N_c$. Pour cela, nous utilisons un paramètre de qualité $\alpha \in [0,1]$. À l'étape 12, nous construisons la liste RN_c en utilisant le paramètre de sélection α. RN_c regroupe les opérations de N_c qui ont une valeur de leur fonction gloutonne supérieure ou égale à $(g_{\max} - \alpha * (g_{\max} - g_{\min}))$. Le paramètre $\alpha \in [0,1]$ correspond à l'importance du facteur aléatoire de l'algorithme : si $\alpha = 0$, l'algorithme est totalement glouton, si $\alpha = 1$, l'algorithme est totalement aléatoire :

$$RN_c = \{i \in N_c \,|\, g_i \geq g_{\max} - \alpha * (g_{\max} - g_{\min})\}.$$

Une opération i est ensuite sélectionnée aléatoirement de la liste RN_c et affectée à la station courante. Toutes les opérations de $LI(i)$ y sont ajoutées (étape 12).

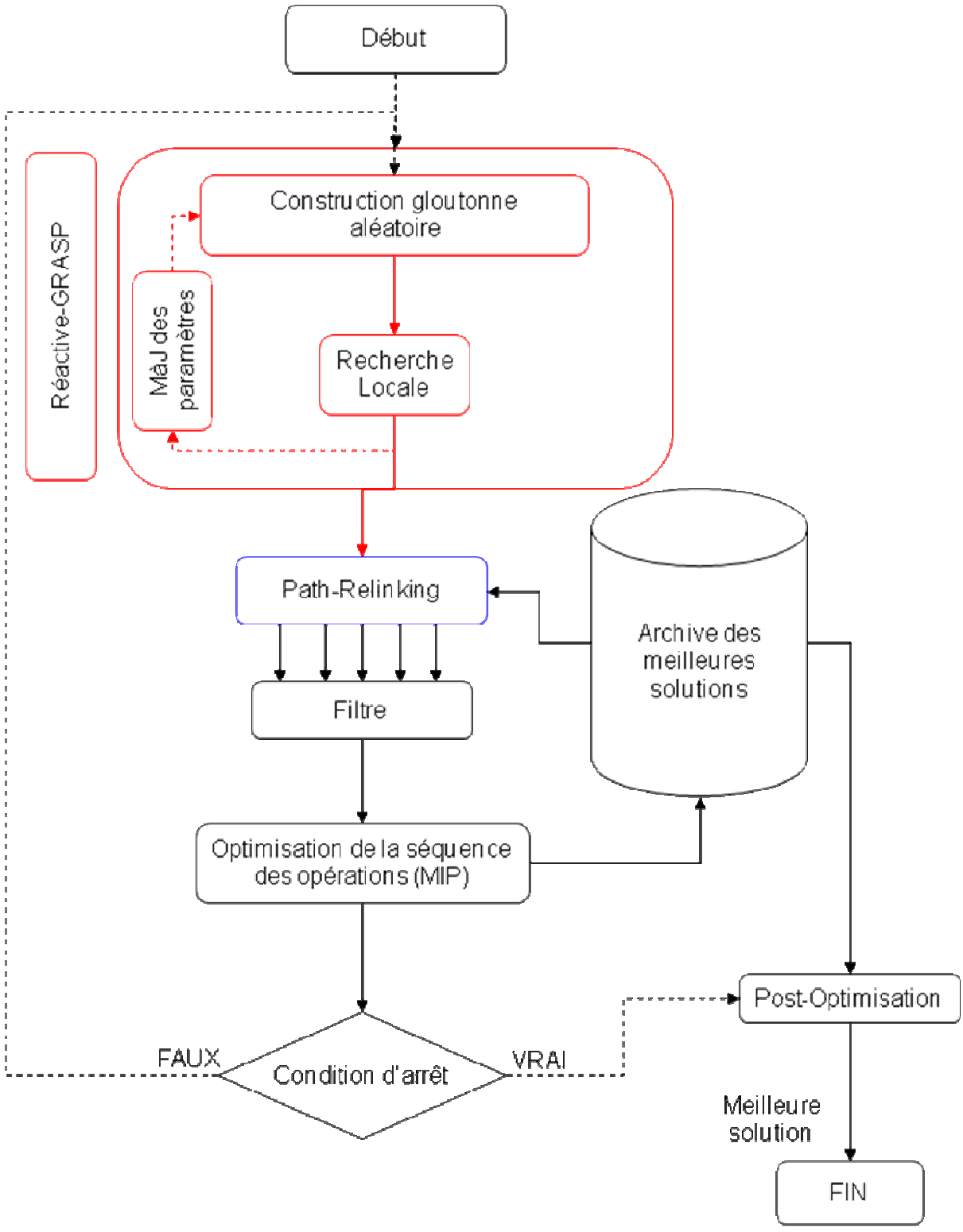

Figure 4.1 Schéma général de l'approche basée sur GRASP

4.3.1.2. Phase de recherche locale

La phase de recherche locale consiste à explorer les voisinages des solutions générées à la phase de construction pour essayer d'améliorer leur qualité. Nous considérons quatre voisinages différents qui sont explorés successivement (V_1, V_2, V_3 et V_4). En explorant ces

voisinages, nous cherchons à améliorer la qualité des solutions en minimisant le nombre de CU et de stations de travail, en améliorant l'équilibrage de charge dans la ligne et en réduisant les temps de setup en optimisant la séquence des opérations pour les stations. La recherche locale est basée sur un algorithme de descente qui fait appel itérativement à une procédure de recherche pour chaque voisinage. Tous les voisinages V_1, V_2, V_3 et V_4 sont étudiés en utilisant la même stratégie de recherche. Cette stratégie consiste à choisir la première amélioration trouvée, c.-à-d., le premier voisin trouvé qui a une meilleure qualité que la solution courante est sélectionnée.

Algorithme 4.3 Algorithme semi-glouton

Début

1: $k \leftarrow 1, N_a \leftarrow N, N_r \leftarrow \emptyset$
2: **Tant que** $(N_a \neq \emptyset)$ **Faire**
3: Construire N_c
4: **Si** $(N_c = \emptyset)$ **Alors**
5: $k \leftarrow k+1$
6: **Sinon**
7: **Pour** $(i = 1,...,|N_c|)$ **Faire**
8: Construire la liste $LI(i)$
9: **Fin pour**
10: $g_{\max} \leftarrow \max_{\{i \in N_c\}}(g(i))$
11: $g_{\min} \leftarrow \min_{\{i \in N_c\}}(g(i))$
12: $RN_c \leftarrow \{i \in N_c | g(i) \geq g_{\max} - \alpha * (g_{\max} - g_{\min})\}$
13: Sélectionner aléatoirement un élément $i \in RN_c$
14: Affecter *i* à la station *k* et mettre à jour $N(k)$
15: **Si** $(T_s(k) < 0)$ **Alors**
16: **Si** $(M(k) < n_0)$ **Alors**
17: $M(k) \leftarrow M(k)+1$, Mettre à jour N_a, N_r, $T_s(k)$
18: **Sinon** $N(k) \leftarrow N(k) \setminus LI(i) \setminus$, $\overline{N_c} \leftarrow \overline{N_c} \cup \{i\}$
19: **Fin Si**
20: **Fin Si**
21: **Fin Si**
22: **Fin Tant que**

Fin

Pour évaluer les différentes solutions, nous utilisons la fonction $coût(s)$ qui correspond à la valeur de la fonction objectif de la solution s. Pour chaque solution obtenue par l'algorithme de construction semi-glouton, nous appliquons la procédure de recherche locale. Notre

objectif est de minimiser le coût de la solution, ainsi, si le coût d'une solution voisine est inférieur de celui de la solution courante, nous remplaçons la solution courante par la solution voisine. Nous appliquons les procédures de recherche dans différents voisinages de façon séquentielle et quand nous obtenons une solution améliorée, nous réappliquons ces procédures. La recherche dans les voisinages est effectuée par différents algorithmes (*Recherche_*$V_n(s)$, $\forall n \in \{1,2,3,4\}$). L'Algorithme 4.4 résume l'approche de recherche locale.

Algorithme 4.4 Recherche locale

Début

1: $s^* \leftarrow s$
2: $Test\_amélioration \leftarrow 1$
3: **Tant que** $(Test\_amélioration = 1)$ **Faire**
4: $\quad Test\_amélioration \leftarrow 0$
5: $\quad$ **Pour** $(n = 1,...,4)$ **Faire**
6: $\quad\quad Recherche\_V_n(s^*)$
7: $\quad$ **Fin Pour**
8: $\quad$ **Si** $(Coût(s^*) < Coût(s))$ **Alors**
9: $\quad\quad s \leftarrow s^*$
10: $\quad\quad Test\_amélioration \leftarrow 1$
11: $\quad$ **Fin Si** ;
12: **Fin Tant que** ;

Fin

Dans la suite, nous détaillons les différents voisinages et les algorithmes de recherche correspondants.

4.3.1.2.1. Minimiser le nombre de CU

La première recherche locale est conçue pour la recherche de solutions avec le moins de CU possible. Afin d'atteindre cet objectif, nous considérons le domaine de voisinage V_1 qui contient toutes les solutions qui peuvent être obtenues en déplaçant un sous-ensemble d'opérations d'une station à une autre. Ceci peut permettre de diminuer le nombre de CU en allégeant la charge d'une station jusqu'à la suppression d'un CU sans ajouter d'autres CU sur les autres stations.

$$V_1(s) = \left\{ s' \;\middle|\; \begin{array}{l} s'(N_{WS}) = s(N_{WS}), \exists k \in \{1,...,s(N_{WS})\}, s'(N(k)) \subset s(N(k)) \\ \quad et\ \forall k \in \{1,...,s(N_{WS})\}, k' \neq k, s(N(k)) \subseteq s'(N(k)) \\ \quad et \sum_{1 \le k \le s'(N_{WS})} s'(M(k)) < \sum_{1 \le k \le s(N_{WS})} s(M(k)) \end{array} \right\}$$

Le schéma de l'algorithme de recherche du premier voisinage est présenté dans l'Algorithme 4.5. Cet algorithme vérifie pour une station de travail s'il est possible de réaffecter quelques unes de ses opérations à une autre station dans le but de supprimer un ou plusieurs CU. La réaffectation des opérations doit respecter les contraintes du problème. Après cela, si le nombre de CU diminue, la nouvelle solution est enregistrée et remplace la solution courante, sinon, l'algorithme passe à la station suivante. Cette procédure est répétée jusqu'à ce que toutes les stations soient visitées. Cet algorithme peut affecter la qualité de l'équilibrage de la charge qui n'est pas prise en compte ici.

Algorithme 4.5 $Recherche\_V_1(s)$

Début

1: $s' \leftarrow s$
2: **Pour** $(k = 1,..., s'(N_{WS}))$ **Faire**
3: $m \leftarrow s'(M(k))$
4: **Pour** $(i = 1,..., |s'(N(k))|)$ **Faire**
5: **Pour** $(k' = 1,..., s'(N_{WS}))$ **Faire**
6: $s''(N(k')) \leftarrow s'(N(k')) \cup \{s'(N(k)(i))\}$
7 $s''(N(k)) \leftarrow s'(N(k)) \cup \{s'(N(k)(i))\}$
8: **Si** $(s'' est\ admissible)$ **Alors**
9: $s'(N(k')) \leftarrow s''(N(k'))$
10: $s'(N(k)) \leftarrow s''(N(k))$
11: Mettre à jour $s'(M(k)) et\ s'(M(k'))$
12: **Fin Si**
13: **Fin Pour**
14: **Fin Pour**
15: **Si** $(m > s'(M(k)))$ **Alors**
16: $s \leftarrow s'$
17: **Sinon**
18: $s' \leftarrow s$
19: **Fin Si**
20: **Fin Pour**

Fin

4.3.1.2.2. Minimiser le nombre de stations de travail

La deuxième recherche locale est conçue pour réduire le nombre de stations. Cet objectif est atteint grâce à l'exploration du voisinage V_2. Ce voisinage contient toutes les solutions qui peuvent être obtenues par la réaffectation de toutes les opérations d'une ou plusieurs stations aux autres.

$$V_2(s) = \left\{ s' \;\middle|\; \begin{array}{l} s'(N_{WS}) < s(N_{WS}) \\ \quad et\, \exists k \in \{1,\ldots,s(N_{WS})\}, \forall k' \in \{1,\ldots,k-1\}, s(N(k')) \subseteq s'(N(k)) \\ \quad et\, \forall k' \in \{k,\ldots,s(N_{WS})-1\}, s(N(k'+1)) \subseteq s'(N(k')) \\ \quad et \sum_{k \in s'(N_{WS})} s'(M(k)) \leq \sum_{k \in s(N_{WS})} s(M(k)) \end{array} \right\}$$

Algorithme 4.6 *Recherche*$\_V_2(s)$

Début

1: $s' \leftarrow s$

2: **Pour** $(k = 1,\ldots, s'(N_{WS}))$ **Faire**

3: $m \leftarrow s'(M(k))$

4: **Pour** $(i = 1,\ldots, |s'(N(k))|)$ **Faire**

5: **Pour** $(k' = 1,\ldots, s'(N_{WS}))$ **Faire**

6: $s'' \leftarrow s' \;\middle|\; \begin{array}{l} s'(N(k') \leftarrow s'(N(k') \cup \{s'(N(k)(i))\} \\ et \\ s'(N(k) \leftarrow s'(N(k) \setminus \{s'(N(k)(i))\} \end{array}$

7 **Si** $(s''$ *est admissible*$)$ **Alors**

8: $s'(N(k')) \leftarrow s'(N(k')) \cup \{s'(N(k)(i))\}$

9: $s'(N(k)) \leftarrow s'(N(k)) \setminus \{s'(N(k)(i))\}$

10: **Fin Si**

11: **Fin Pour**

12: **Fin Pour**

13: **Si** $\left(s'(N(k)) = \emptyset \text{ et } \sum_{k=1}^{k=s'(N_{WS})} s'(M(k)) < \sum_{k=1}^{k=s(N_{WS})} s(M(k)) \right)$ **Alors**

14: **Pour** $(k' = k,\ldots, s'(N_{WS}) - 1)$ **Faire**

15: $s'(N(k')) \leftarrow s'(N(k'+1))$

16: **Fin Pour**

17: $s'(N_{WS}) \leftarrow s'(N_{WS}) - 1$

18: $s \leftarrow s'$

19: **Sinon**

20: $s' \leftarrow s$

21: **Fin Si** ;

22: **Fin Pour** ;

Fin

En effet, une station de travail peut être supprimée en réaffectant toutes ses opérations aux autres stations. L'Algorithme 4.6 permet de vérifier les stations une par une et cherche à

réaffecter toutes leurs opérations aux autres stations. Si le nombre total de CU dans une solution voisine n'augmente pas, nous remplaçons la solution courante par cette nouvelle solution.

4.3.1.2.3. Amélioration de la qualité d'équilibrage de la charge

La troisième recherche locale est conçue afin d'améliorer l'équilibrage de la charge. Ceci permettra entre autre de minimiser le temps de cycle par le déplacement d'opérations des stations les plus chargées aux stations les moins chargées de la ligne. Elle s'appuie sur le voisinage V_3 qui contient les solutions avec le même nombre de stations mais une affectation différente des opérations.

$$V_3(s) = \left\{ s' \left| \begin{array}{l} s'(N_{WS}) = s(N_{WS}) \\ \quad et\ \exists k, k' \in \{1, \ldots, s(N_{WS})\}, s(t_{ut}(k)) > s(\tilde{t}_{ut}),\ s(t_{ut}(k')) < s(\tilde{t}_{ut}) \\ \quad\quad s'(N(k)) \subset s(N(k)), s(N(k')) \subset s'(N(k')) \\ \quad et\ \forall k'' \neq k, k', s(N(k'')) = s'(N(k'')) \end{array} \right. \right\}$$

L'Algorithme 4.7 recherche les stations de travail dont la charge est supérieure à la charge moyenne de la ligne et ensuite cherche à alléger la charge de ces stations par la réaffectation d'opérations aux stations dont la charge est inférieure à la charge moyenne.

4.3.1.2.4. Optimisation de la séquence d'opérations

Un quatrième voisinage consiste à rechercher les solutions voisines dont les séquences d'opérations sont mieux ordonnées (V_4). Puisque le temps opératoire d'une station dépend de la séquence de ses opérations (à cause des temps inter-opératoires), optimiser leur séquence revient à diminuer le temps de travail de la station. Nous effectuons la recherche de voisinage V_4 chaque fois qu'on lance la procédure de recherche locale. Trois méthodes de recherche ont été développées pour ce voisinage, à savoir : deux algorithmes heuristiques, l'un glouton, le second se basant sur des choix aléatoire, ainsi qu'un programme linéaire en nombres mixtes (MIP), pour une optimisation exacte de la séquence.

Algorithme 4.7 *Recherche_V*$_3(s)$

Début

1: $s' \leftarrow s$

2: **Pour** $(k = 1,..., s'(N_{WS}))$ **Faire**

3: **Si** $(s'(t_{ut}(k)) > s'(\tilde{t}_{ut}))$ **Alors**

4: $k' \leftarrow 1$

5: **Tant que** $((k' < s'(N_{WS}))\, et\, (Test\_imp = 0))$ **Faire**

6: **Si** $(s'(t_{ut}(k')) < s'(\tilde{t}_{ut}))$ **Alors**

7 **Pour** $(i = 1,..., |s'(N(k))|)$ **Faire**

8:
$$s'' \leftarrow s' \begin{cases} s'(N(k')) \leftarrow s'(N(k')) \cup \{s'(N(k)(i))\} \\ et \\ s'(N(k)) \leftarrow s'(N(k)) \setminus \{s'(N(k)(i))\} \end{cases}$$

9: **Si** $(s''\, est\, admissible)$ **Alors**

10: $s'(N(k')) \leftarrow s'(N(k')) \cup \{s'(N(k)(i))\}$

11: $s'(N(k)) \leftarrow s'(N(k)) \setminus \{s'(N(k)(i))\}$

12: Mettre à jour $s', Test\_imp \leftarrow 1, k' \leftarrow k'+1$

13: **Fin Si**

14: **Fin Pour**

15: **Fin Si**

16: **Fin Tant que**

17: **Fin Si**

18: **Fin Si**

19: $s \leftarrow s'$

Fin

Algorithme 4.8 *Recherche*$\_V_4(s)$

Début

1: $s' \leftarrow s, i \leftarrow 1$

2: **Pour** $(k = 1,..., s'(N_{WS}))$ **Faire**

3: **Si** $(\frac{T_0 - s'(Ts(k))}{s'(M(k)) * T_0} < IMP\_Rate\_MIP)$ **Alors**

4: $Optimiser\_séquence\_MIP(s'(N(k)))$

5: **Sinon**

6: $Optimiser\_séquence\_Glouton(s'(N(k)))$

7 **Si** $(s'(Ts(k)) > s(Ts(k)))$ **Alors**

8: $s \leftarrow s'$

9: **Sinon**

10: $s' \leftarrow s$

11: **Fin Si**

12: *Recherche_voisin* $(s'(N(k)))$

13: **Fin Si**

14: **Fin Pour**

Fin

Dans l'Algorithme 4.8, nous expliquons la procédure de recherche à l'aide de ces trois méthodes. L'utilisation du MIP est soumise à une condition d'amélioration possible (suppression d'un CU). L'introduction de cette condition est liée au temps de calcul non négligeable pour résoudre le MIP. Donc, pour avoir un tel critère, à chaque fois que le MIP est utilisé, le niveau d'amélioration est enregistré (*IMP – Rate – MIP* , variation de la charge de la station, ou niveau d'amélioration). En utilisant cette mémoire, nous pouvons estimer si l'utilisation du MIP peut supprimer ou non un CU ou plus sur une station de travail en considérant le temps de cycle propre à la station et le nombre de CU dans la solution courante. L'approche MIP sera détaillée dans la section suivante. Ce paramètre est mis à jour à chaque utilisation du MIP.

Algorithme 4.9 Optimisation d'une séquence en utilisant un algorithme glouton

Début

1: $N_k \leftarrow s'(N(k)), l \leftarrow 1$

2: Sélectionner $i \in N_k \left| t_i = \min_{j \in N_k}(t_j) et P_i^* \cap N_k = \emptyset \right.$

3: $s'(N(k)(l)) \leftarrow i, N_k \leftarrow N_k \setminus \{i\}$

4: $l \leftarrow l+1, Op \leftarrow i$

5: **Tant que** $(N_k \neq \emptyset)$ **Faire**

6: Sélectionner $i \in N_k \left| t_{Op,i} = \min_{j \in N_k}(t_{Op,j}) et P_i^* \cap N_k = \emptyset \right.$

7: $s'(N(k)(l)) \leftarrow i, N_k \leftarrow N_k \setminus \{i\}$

8: $l \leftarrow l+1, Op \leftarrow i$

9: **Fin Tant que**

Fin

Algorithme 4.10 Recherche voisinage *N*(*k*)

Début

1: $Ts \leftarrow Ts(k)$

2: **Pour** $(i = 1,...,|N(k)|)$ **Faire**

3: Déplacer $N(k)(i)$ à une autre position

4: Calculer $Ts(k)$

5: **Si** $(Ts(k) < Ts)$ **Alors**

6: Conserver $N(k)$

7: **Fin Si**

8: **Fin Pour**

Fin

Dans l'Algorithme 4.8, nous utilisons également deux approches heuristiques détaillées dans les Algorithmes 4.9. et 4.10. La première approche consiste à réorganiser la séquence d'opérations suivant l'ordre décroissant des temps inter-opératoires en respectant les contraintes de précédence du problème. Si la nouvelle séquence est meilleure que la séquence initiale, elle est sauvegardée. La seconde approche consiste à réaffecter les opérations au hasard jusqu'à une amélioration possible tout en respectant les contraintes de précédence. Les trois méthodes (MIP et les deux heuristiques) sont utilisées de façon séquentielle.

4.3.1.3. Approche MIP pour l'optimisation de la séquence d'opérations

Nous avons défini un programme linéaire en nombres mixtes pour une optimisation exacte de la séquence d'opérations sur une station de travail. L'utilisation de la programmation linéaire pour la résolution d'un sous-problème dans une métaheuristique est assez récente dans la littérature. Le nouveau concept de matheuristique a été défini comme la combinaison de métaheuristiques avec la programmation mathématique (voir par exemple : Guschinskaya et al., 2008 ; Hanafi et al., 2009). Nous considérons les stations de travail une par une, et nous cherchons à optimiser l'ordonnancement des opérations en appliquant le MIP. En prenant en compte les temps inter-opératoires et les contraintes de précédence, le problème peut être considéré comme un problème d'ordonnancement d'une machine avec des contraintes de précédence et des temps inter-opératoires (voir Yalaoui et Chu, 2002). Dans Bigras et al. (2008), les auteurs ont montré que ce problème est équivalent à un problème du voyageur de commerce avec dépendance entre les temps de déplacement.

Nous optons pour une formulation en nombres mixtes inspirée du modèle de voyageur de commerce dénommé DL et proposée par Desrochers et Laporte (1991). Dans la formulation que nous proposons, nous notons deux différences majeures avec la formulation DL, à savoir : i) le graphe considéré n'est pas hamiltonien et ii) nous prenons en compte les contraintes de précédence. Ce modèle est potentiellement applicable à toutes les solutions avant leur archivage. Il est utilisé aussi pour la recherche locale si d'après notre estimation son application peut permettre la suppression de CU.

Nous utilisons quelques notations supplémentaires :

n : le nombre d'opérations dans la séquence ;

t_{ij} : coût d'un déplacement de i à j (temps inter-opératoire).

Les variables de décision sont les suivantes :

$x_{ij} = 1$ si la séquence (i , j) est sélectionnée, sinon, $x_{ij} = 0$;

u_i : la position de l'opération i dans la séquence $N(k)$;

$f_i = 1$ si l'opération i est affectée à la première position de la séquence, sinon, $f_i = 0$;

$l_i = 1$ si l'opération i est affectée à la dernière position de la séquence, sinon, $l_i = 0$.

Le programme linéaire en nombres mixtes est comme suit :

- la fonction objectif (4.1) minimise le temps de chargement de la station de travail :

$$\text{Minimiser} \sum_{i=1}^{n} \sum_{j=1}^{n} t_{ij} * x_{ij} \tag{4.1}$$

- Les équations (4.2-4.5) assurent qu'une position n'est occupée que par une et une seule opération :

$$\sum_{i=1, i\neq j}^{n} x_{ij} = 1 - f_j, \forall j = 1,2,\ldots,n \tag{4.2}$$

$$\sum_{j=1, j\neq i}^{n} x_{ij} = 1 - l_i, \forall j = 1,2,\ldots,n \tag{4.3}$$

$$\sum_{j=1}^{n} f_j = 1 - f_j \tag{4.4}$$

$$\sum_{j=1}^{n} l_j = 1 \tag{4.5}$$

- Les équations (4.6-4.9) assurent qu'il n'y aura qu'une seule séquence :

$$u_i + (n-1) * f_i \leq n, \forall i \in \{1,...,n\}, et\, i \neq j \tag{4.6}$$

$$u_i + (n-1) * l_i \geq 1, \forall i \in \{1,...,n\}, et\, i \neq j \tag{4.7}$$

$$u_i - u_j + n * x_{ij} + (n-2) * x_{ji} \leq n-1, \forall i, j \in \{1,...,n\}, et\, i \neq j \tag{4.8}$$

$$u_i - u_j - (n-2) * x_{ij} - n * x_{ji} \geq 1-n, \forall i, j \in \{1,...,n\}, et\, i \neq j \tag{4.9}$$

- L'équation (4.10) définit les contraintes de précédence entre les opérations :

$$1 + u_j \leq u_i, \forall i \in \{1,...,n\}, \forall j \in P_i^* \tag{4.10}$$

- Les équations (4.11-4.14) fournissent des contraintes additionnelles pour les valeurs possibles des variables :

$$x_{ij} \in \{0,1\}, \forall i, j \in \{1,...,n\} \tag{4.11}$$

$$f_i \in \{0,1\}, \forall i \in \{1,...,n\} \tag{4.12}$$

$$l_i \in \{0,1\}, \forall i \in \{1,...,n\} \tag{4.13}$$

$$u_i \in \Re, \forall i \in \{1,...,n\} \tag{4.14}$$

4.3.2. Reactive GRASP

Plusieurs améliorations ont été apportées dans la littérature au schéma de base de la méthode GRASP. Parmi celles-ci, le Reactive GRASP. Cette amélioration consiste à calibrer automatiquement le paramètre de sélection aléatoire α. Dans Feo et Resende (1989), les auteurs ont déjà discuté le choix du paramètre α pour améliorer la qualité des solutions et leur diversité. Pour un jeu de données différent, des valeurs différentes de α peuvent amener à des meilleurs résultats. Une possibilité de surmonter cette difficulté est l'utilisation d'une procédure d'auto-adaptation qui permet de définir la valeur de α de façon dynamique, c'est l'idée du Reactive-GRASP. Le Reactive-GRASP (Prais et Ribeiro, 2000, Delmaire et al., 1999) ajuste de façon autonome la valeur de α suivant la qualité des solutions générées au cours de la résolution.

Nous définissons les paramètres suivants :

- m : nombre des valeurs possibles de α ;
- $A_\alpha = \{\alpha_1, ..., \alpha_m\}$: liste des valeurs possibles de α ;
- p_i : probabilité de choisir α_i lors de la sélection d'une valeur de α ;
- A_i : coût moyen des solutions générées avec $\alpha = \alpha_i$;

La procédure permet de sélectionner une valeur de α à partir d'une liste $A_\alpha = \{\alpha_1, ..., \alpha_m\}$ contenant m valeurs acceptables et prédéterminées de α. L'utilisation de différentes valeurs de α permet la construction de différentes listes restreintes RN_c, ce qui peut conduire à construire des solutions différentes. Un autre paramètre δ est utilisé pour calibré la procédure d'ajustement. Pour chaque valeur possible de α ($\alpha_i, i \in \{1, ..., m\}$), une probabilité p_i est définie. Les probabilités p_i sont initialisées selon une distribution uniforme ($p_i = 1/m, \forall i \in \{1, ..., m\}$). Une fois la procédure de résolution lancée, ces probabilités sont mises à jour périodiquement en utilisant les informations collectées lors des itérations précédentes. Après un nombre fixe d'itérations, le coût moyen A_i des solutions obtenues avec $\alpha = \alpha_i$ est calculé. Les nouvelles valeurs de probabilités sont calculées en tenant en compte de la qualité moyenne des solutions obtenues avec α_i comparée à la meilleure solution générée par la procédure à cet instant de la résolution. Plus petite est la valeur de A_i, plus grande sera la valeur de p_i correspondante. En conséquence, durant les itérations suivantes,

les valeurs de α qui mènent aux meilleures solutions ont une probabilité plus forte d'être sélectionnées.

Algorithme 4.11 Reactive GRASP

Début

1: $Coût(\hat{s}) \leftarrow \infty$

2: $\forall i \in \{1,...,m\}, p_i \leftarrow \frac{1}{m}, sum_i \leftarrow 0, n_i \leftarrow 0$

3: **Pour** $(it = 1,...,it.\max)$ **Faire**

4: Sélectionner $\alpha_i \in A_\alpha$ selon la probabilité p_i

5: $n_i \leftarrow n_i + 1$

6: $s \leftarrow Algorithme\_Glouton\_Aléatoire(\alpha_i)$

7 $s' \leftarrow recherche\_locale(s)$

8: **Si** $(Coût(s') < Coût(\hat{s}))$ **Alors**

9: $s^* \leftarrow s'$

10: $Coût(\hat{s}) \leftarrow Coût(s^*)$

11: **Fin Si**

12: $sum_i \leftarrow sum_i + Coût(s')$

13: $A_i \leftarrow \frac{sum_i}{n_i}$

14: $q_i \leftarrow \left(\frac{Coût(s^*)}{A_i - Coût(s^*) + 1} \right)^\delta$

15: $p_i \leftarrow \frac{q_i}{\sum_{j=1}^{j=m} q_j}$

16: **Fin Pour**

17: Récupérer s^*

Fin

4.3.3. Procédure d'intensification (Path-Relinking)

Pour finir, nous présentons la procédure d'intensification que nous avons utilisée. Cette procédure s'appuie sur la technique du Path-Relinking (recomposition de chemin). Le Path-Relinking consiste à explorer le chemin entre deux solutions différentes. Chaque chemin lie une solution de départ (starting solution, x_s) à une solution d'arrivée (target solution, x_t). Cette liaison est assurée par des déplacements des éléments de la solution de départ vers leur emplacement dans la solution d'arrivée. Le Path-Relinking est une stratégie qui cherche à incorporer les attributs des solutions de haute qualité.

Nous utilisons une liste des différences symétriques entre les deux solutions $Diff(x, y)$ (de taille $\Delta(x, y)$). Cette liste correspond aux mouvements nécessaires pour que la solution x atteigne y et elle est mise à jour à chaque mouvement effectué. Une liste de solutions intermédiaires (qui correspondent à la connexion entre les deux solutions) est ainsi générée. Ces solutions peuvent être non admissibles, nous avons donc ajouté une procédure de réparation de solutions. Elle consiste à chercher les incohérences par rapport aux contraintes du problème et à les réparer. Nous fixons un nombre maximum des déplacements à effectuer. L'exploration de la trajectoire s'effectue alors afin de générer des solutions de meilleure qualité à partir des deux solutions (de départ et d'arrivée).

Algorithme 4.12 Déplacement (*x*,*y*)

Début

1: Calculer $Diff(x, y), \Delta(x, y)$

2: $test \leftarrow 0, x' \leftarrow x$

3: **Pour** $(i = 1, ..., \Delta(x, y))$ **Faire**

4: **Si** $(test = 0)$ **Alors**

5: $Op = Diff(x, y)(i)$ // opération à la $i^{ème}$ position de $Diff(x, y)$ //

6: $k \leftarrow x(St(Op)), k' \leftarrow y(St(Op))$

7 $x'(N(k)) \leftarrow x'(N(k)) \setminus \{Op\}$

8: $x'(N(k')) \leftarrow x'(N(k')) \cup \{Op\}$

9: **Si** $(x' est\ admissible)$ **Alors**

10: $test \leftarrow 1$

11: **Fin Si**

12: **Fin Si**

13: **Fin Pour**

14: Récupérer x'

Fin

Les déplacements sont effectués grâce à l'Algorithme 4.12. qui permet d'effectuer un seul mouvement à chaque appel. L'algorithme commence par le calcul du nombre de différences entre les deux solutions et construit une liste $Diff(x, y)$. Les différences correspondent à l'affectation d'une opération à une station (si dans les deux solutions, une opération n'est pas affectée à la même station, cela correspond à une différence et donc à un déplacement). Les déplacements consistent tout simplement à affecter une opération de la solution de départ à son emplacement dans la solution d'arrivée. La sélection des opérations à déplacer concerne en premier lieu les déplacements qui génèrent des solutions admissibles. Si de tels déplacements n'existent pas dans $Diff(x, y)$, alors nous choisissons un déplacement au

hasard. Dans ce cas, la solution générée est non admissible ce qui nous oblige à appliquer la procédure de réparation.

La procédure du Path-Relinking est comme suit (Algorithme 4.13) : nous considérons deux solutions différentes x et y et nous calculons le nombre de différences entre eux. Si ce nombre est non nul et que la condition d'arrêt (nombre maximum de mouvements) n'est pas vérifiée, nous appliquons l'algorithme Deplacement(x,y). Une nouvelle solution est alors générée. Nous vérifions ainsi si cette solution est admissible ou non. Si elle n'est pas admissible, nous vérifions les anomalies (contraintes non respectées) et nous les corrigeons une par une jusqu'à ce que nous obtenions une solution admissible. La qualité de la nouvelle solution est testée afin de savoir si elle doit être ajoutée à l'archive des meilleures solutions ou pas.

Plusieurs stratégies peuvent être définies pour l'implémentation du Path-Relinking (Resende et Ribeiro, 2006). Dans l'approche que nous proposons, nous avons opté pour une application périodique du Path-Relinking, la meilleure solution est considérée comme solution de départ, et l'autre comme solution d'arrivée (backward relinking). L'exploration de la trajectoire entre les deux solutions n'est pas complète, nous nous limitons à la première partie de la trajectoire (truncated relinking). Et finalement, une phase de post optimisation est ajoutée, c.-à-d., le Path-Relinking est appliqué à la fin de toutes les itérations sur toutes les paires de solutions possibles de l'archive.

Algorithme 4.13 Connecter(x,y)

Début

Calculer $\Delta(x, y)$

$\Delta \leftarrow \Delta(x, y)$

Tant que $((condition\_Arrêt = Faux)\, et\, (\Delta \neq 0))$ **Faire**

  $x' \leftarrow Lier(x, y)$

  Calculer $\Delta(x', y)$

  $\Delta \leftarrow \Delta(x', y)$

  Si $(x'\, n'est\, pas\, admissible)$ **Faire**

    $Réparer(x')$

  Fin Si

Fin Tant que

Fin

4.4. Expérimentations

Afin d'évaluer les performances des différentes méthodes proposées pour la résolution du problème, nous avons généré aléatoirement un ensemble d'instances avec des caractéristiques similaires aux caractéristiques de problèmes rencontrés dans l'industrie. Les instances se caractérisent par le nombre d'opérations de l'ensemble N, les densités D_p, D_{inc}, D_{exc} respectivement, densité du graphe de précédence, d'inclusion et d'exclusion. Les contraintes d'inclusion et d'exclusion sont générées avec des valeurs fixes de leur densité, 5% pour les inclusions et 3% pour les exclusions. La densité du graphe de précédence est fixée à 25%. Le temps de cycle (T_0) varie selon la taille de l'instance. Nous avons généré des instances avec différents nombres d'opérations : 20, 40, 60, 80, 100, 120, 140 et 160. Pour chaque taille, cinq instances différentes ont été générées. Le nombre de CU par station de travail (n_0) est fixé à 3. Dans le Tableau 4.1, nous présentons les valeurs des temps de cycle en fonction de la taille d'instance.

Tableau 4.1 Temps de cycle en fonction du nombre d'opérations

#opérations	20	40	60	80	100	120	140	160
Temps de cycle (unité de temps)	33	66	100	133	166	200	233	266

Nous avons appliqué les différentes approches pour la résolution du problème. Les calculs ont été effectués sur un serveur SUN UltraSPARC IIIi avec 1593 Mhz de CPU et 16 GB de mémoire. Pour les approches itératives, le temps de calcul a été fixé en fonction de nombre d'opérations. Pour chaque instance, chaque algorithme est lancé dix fois sauf pour les algorithmes gloutons qui sont déterministes (un seul lancement). Durant l'exécution de l'algorithme, nous observons l'évolution des résultats obtenus à la fin de chaque intervalle de temps. Ces intervalles sont fixés à un dixième du temps de calcul alloué. Le temps alloué (T_{CPU}) pour différentes instances est fixé à :

$$T_{CPU} = \frac{|N|^2}{5} ;$$

Les paramètres de l'algorithme ACO sont initialisés comme suit :

- α = 8 et décroit à chaque itération ;
- β = 4 et croit à chaque itération.

À chaque itération, les paramètres α et β sont mis à jour suivant les formules suivantes :

$\alpha \leftarrow \alpha * 0{,}99$

$\beta \leftarrow \beta * 1{,}01$

Dans le Tableau 4.2, nous rapportons les résultats obtenus par l'algorithme glouton avec les trois critères de sélection. Nous donnons également la meilleure solution obtenue toutes méthodes confondues qui correspond à une borne supérieure. Nous rapportons aussi une borne inférieure pour chaque instance en utilisant la relaxation linéaire du modèle mathématique présenté dans le Chapitre 3. L'écart en pourcentage entre les meilleurs résultats et la borne inférieure est calculé (colonne Écart). Nous avons alloué 24 heures de calcul par instance pour la relaxation linéaire. Cependant, nous n'avons eu aucun résultat pour les instances de taille supérieure ou égale à 100 opérations (noté N.D.). Nous nous contentons donc de calculer l'écart entre la meilleure solution connue et la borne inférieure pour les instances de taille 20, 40, 60, 80. Les écarts sont compris entre 20% et 40%. Il est difficile de déterminer avec certitude si ces écarts proviennent de la mauvaise qualité de la borne inférieure ou de la meilleure solution connue. Il est cependant intéressant de noter que l'écart observé, sur les instances avec moins de 20 opérations qui ont été résolues par Cplex, entre la borne inférieure et la solution optimale est en moyenne de 21,35%.

En comparant les résultats obtenus grâces aux trois règles de priorité, nous remarquons que la plupart des meilleures solutions sont générées grâce à la règle TCIP et que les solutions de plus mauvaise qualité sont générées par la règle TO. TCIP permet même parfois de générer des solutions de meilleure qualité que celles générées par COMSOAL. Ceci nous conforte dans notre choix d'utiliser cette règle dans ACO et GRASP.

Dans la Figure 4.2, nous présentons les valeurs moyennes des déviations pour les solutions de toutes les instances de chaque taille. Nous remarquons que les valeurs moyennes des solutions générées par GRASP et ACO sont assez proches. L'écart entre leurs déviations moyennes ne dépasse pas 2% pour la plupart de cas (sauf pour les instances de taille 20). Tandis que les solutions générées par COMSOAL semblent de moins en moins bonnes avec un écart moyen égale à 40% pour l'instance de taille 160. Ceci nous montre les limites de la méthode COMSOAL. Dans la suite, nous nous intéressons de plus près aux résultats du GRASP et de l'ACO afin de les comparer.

Tableau 4.2 Résultats heuristiques gloutonnes et bornes

# Op	instance	Heuristique gloutonne			Meilleure solution connue	Borne inférieure	Ecart (%)
		TO	TCIP	TC			
20	1	4025,51	4004,43	4008,63	3358,90	2547,57	31,85%
	2	3686,04	3657,98	3307,93	3156,24	2450,46	28,80%
	3	3665,27	4023,47	3288,02	3158,52	2524,8	25,10%
	4	4390,08	4013,47	4391,27	3366,79	2688,23	25,24%
	5	4020,85	4708,68	4371,81	3364,13	2763,89	21,72%
40	1	3983,45	3984,95	4004,55	3317,04	2613	26,94%
	2	4744,27	3965,55	3972,95	3317,24	2658,73	24,77%
	3	4343,14	3967,47	3994,93	3357,60	2594,26	29,42%
	4	4017,25	3651,15	3629,23	3194,92	2449,69	30,42%
	5	3639,53	3633,05	4003,92	3197,77	2521,72	26,81%
60	1	4733,06	3984,37	4381,04	3536,25	2579,64	37,08%
	2	4734,47	4013,08	4376,91	3563,92	2574,74	38,42%
	3	4372,41	3619,05	3974,72	3526,98	2557,48	37,91%
	4	4342,3	3619,83	4010,89	3566,34	2611,81	36,55%
	5	5086,78	3992,19	3998,29	3567,04	2634,38	35,40%
80	1	4341,61	3989,85	4398,75	3560,35	2615,12	36,14%
	2	5745,46	3947,44	4735,25	3610,86	2743,26	31,63%
	3	4744,44	3977,32	4388,86	3562,07	2634,51	35,21%
	4	4754,99	3618,83	4760,55	3527,74	2571,24	37,20%
	5	5421,87	4375,45	4778,51	3566,80	2579,05	38,30%
100	1	5455,81	4770,90	5135,36	3578,63	N.D.	N.D.
	2	6481,37	4377,30	5111,98	3603,25	N.D.	N.D.
	3	5446,35	3607,50	4401,99	3561,10	N.D.	N.D.
	4	5407,25	3997,80	6155,15	3600,88	N.D.	N.D.
	5	5435,81	4009,96	4391,83	3561,17	N.D.	N.D.
120	1	6476,97	4405,14	5129,60	4253,06	N.D.	N.D.
	2	6148,53	4388,16	4778,94	3599,25	N.D.	N.D.
	3	5782,33	4022,40	4768,83	3558,47	N.D.	N.D.
	4	6485,85	4741,58	5095,66	3891,23	N.D.	N.D.
	5	6830,20	6204,71	6847,02	3986,86	N.D.	N.D.
140	1	8192,33	6900,34	5823,28	4582,16	N.D.	N.D.
	2	7162,38	5825,37	5085,53	3996,65	N.D.	N.D.
	3	7493,10	5115,98	4766,11	3926,90	N.D.	N.D.
	4	5786,73	3611,80	4399,70	3559,01	N.D.	N.D.
	5	5462,20	4401,51	5470,16	3926,46	N.D.	N.D.
160	1	6481,23	5131,34	5476,85	3610,45	N.D.	N.D.
	2	5460,96	4760,26	5101,82	3603,43	N.D.	N.D.
	3	8521,53	4393,52	5825,95	3599,57	N.D.	N.D.
	4	6176,57	5490,02	5089,55	3599,50	N.D.	N.D.
	5	6138,94	4413,95	4767,19	3598,52	N.D.	N.D.

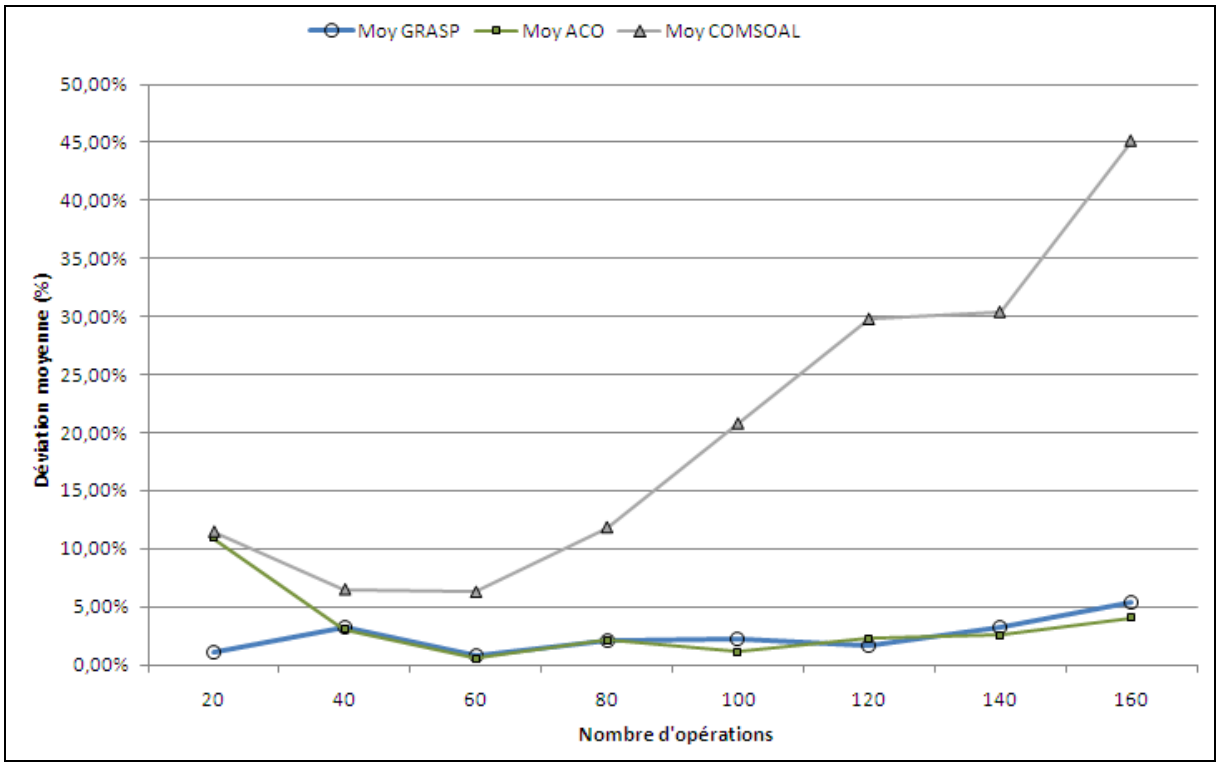

Figure 4.2 Variation des déviations en fonction du nombre d'opérations

Dans la suite, nous allons essayer de faire une comparaison des performances de ces deux méthodes. Afin d'atteindre cet objectif, nous allons utiliser la moyenne des coûts des solutions obtenues par ces deux méthodes. La comparaison des moyennes est une méthode assez utilisée pour une comparaison de deux méthodes itératives même si elle est parfois critiquée. Pour chaque jeu de données de même taille, nous avons calculé la déviation moyenne des solutions générées par le GRASP et l'ACO par rapport à la meilleure solution connue de cette instance. Tous les résultats présentés dans la suite sont exprimés dans cette unité. Afin de remédier aux faiblesses de la méthode de mesure de qualité par comparaison des moyennes, nous avons utilisé en complément le test de Mann-Whitney (Dréo et al., 2003 ; Conover, 1999). Le principe de ce test consiste à répondre à la question : une méthode *A* est elle significativement meilleure qu'une méthode *B*. Ce test consiste à classer par qualité décroissante les solutions générées par deux méthodes différentes et à calculer la somme des rangs des solutions obtenues par une des deux méthodes. Nous pouvons ainsi utiliser cette somme pour calculer les chances qu'une méthode *A* génère des solutions significativement meilleures qu'une méthode *B*. Nous appliquons ce test sur les résultats obtenus pour dix lancements dans des conditions équivalentes de chaque algorithme (cinq jeux de données différents pour chaque taille d'instance) et nous vérifions l'évolution des résultats avec le temps.

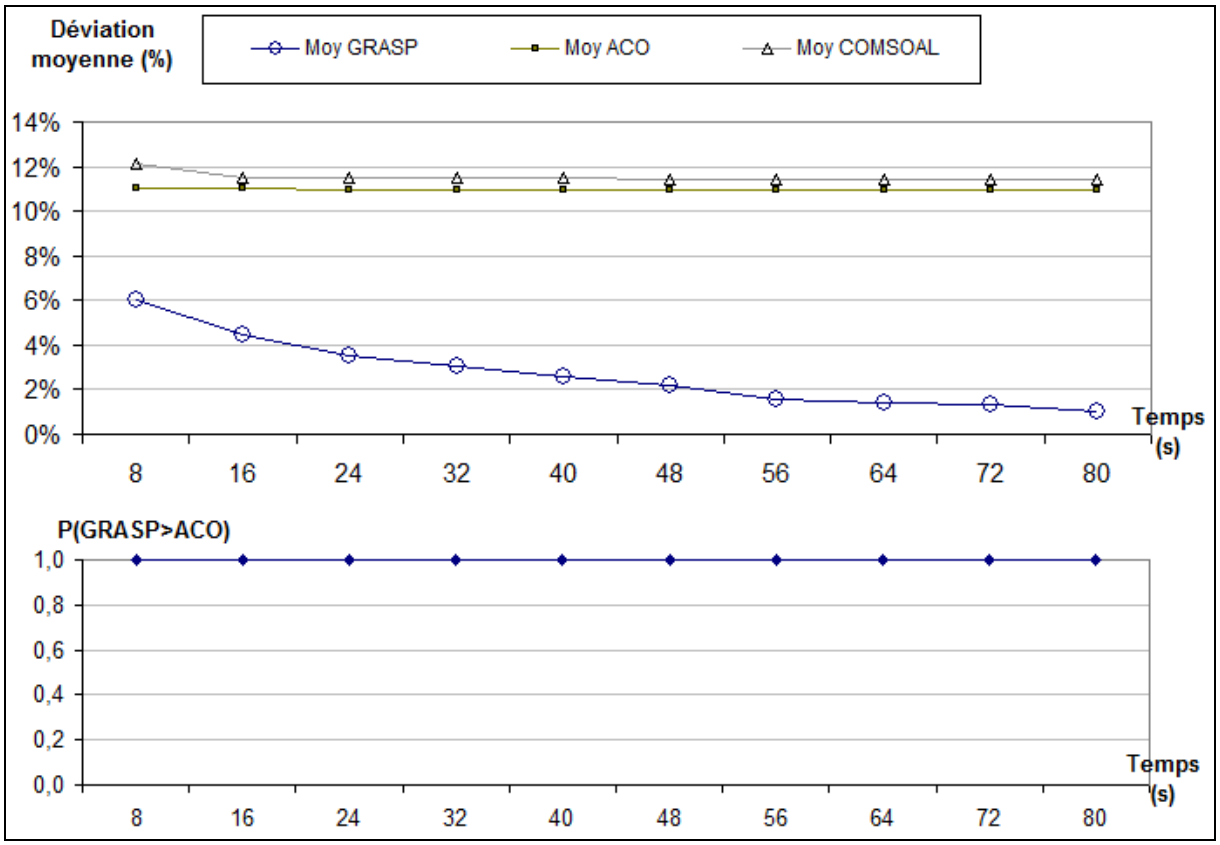

Figure 4.3 Comparaison des heuristiques itératives pour les instances de taille 20

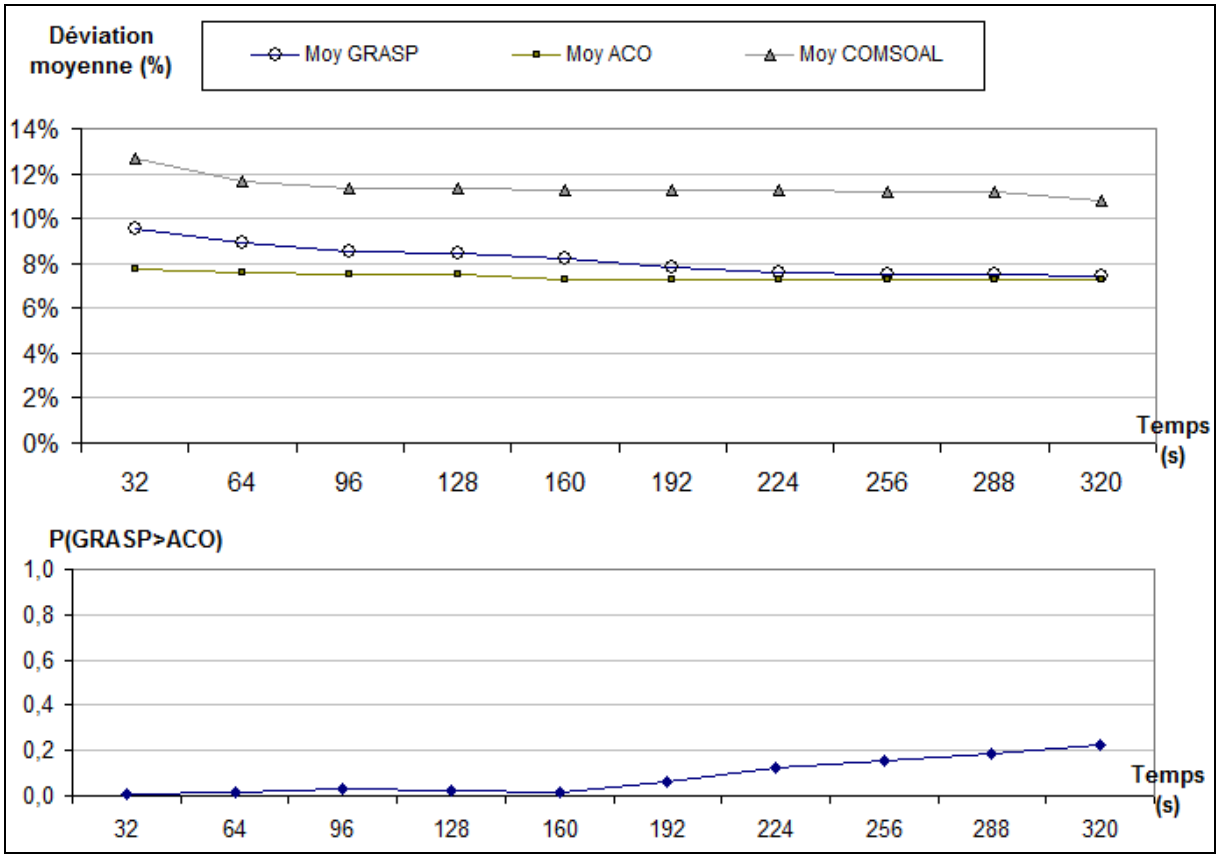

Figure 4.4 Comparaison des heuristiques itératives pour les instances de taille 40

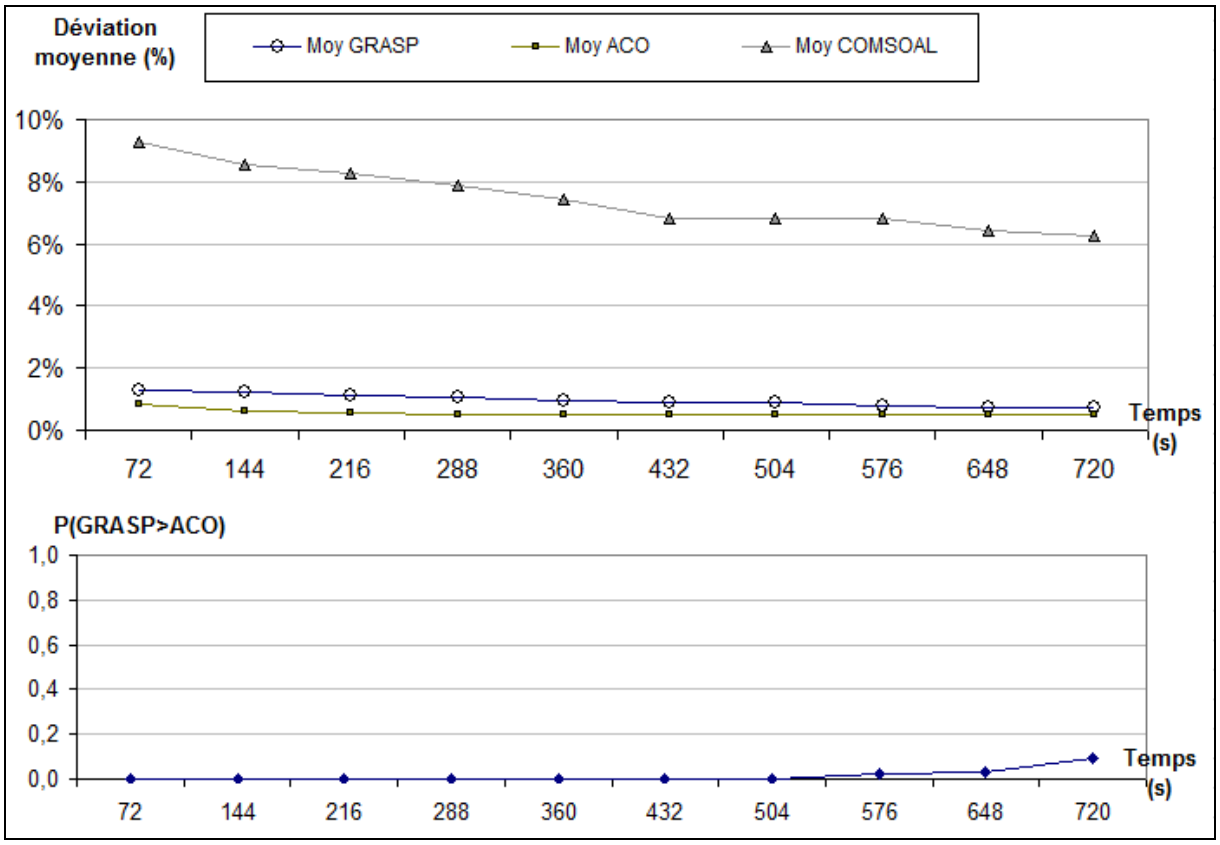

Figure 4.5 Comparaison des heuristiques itératives pour les instances de taille 60

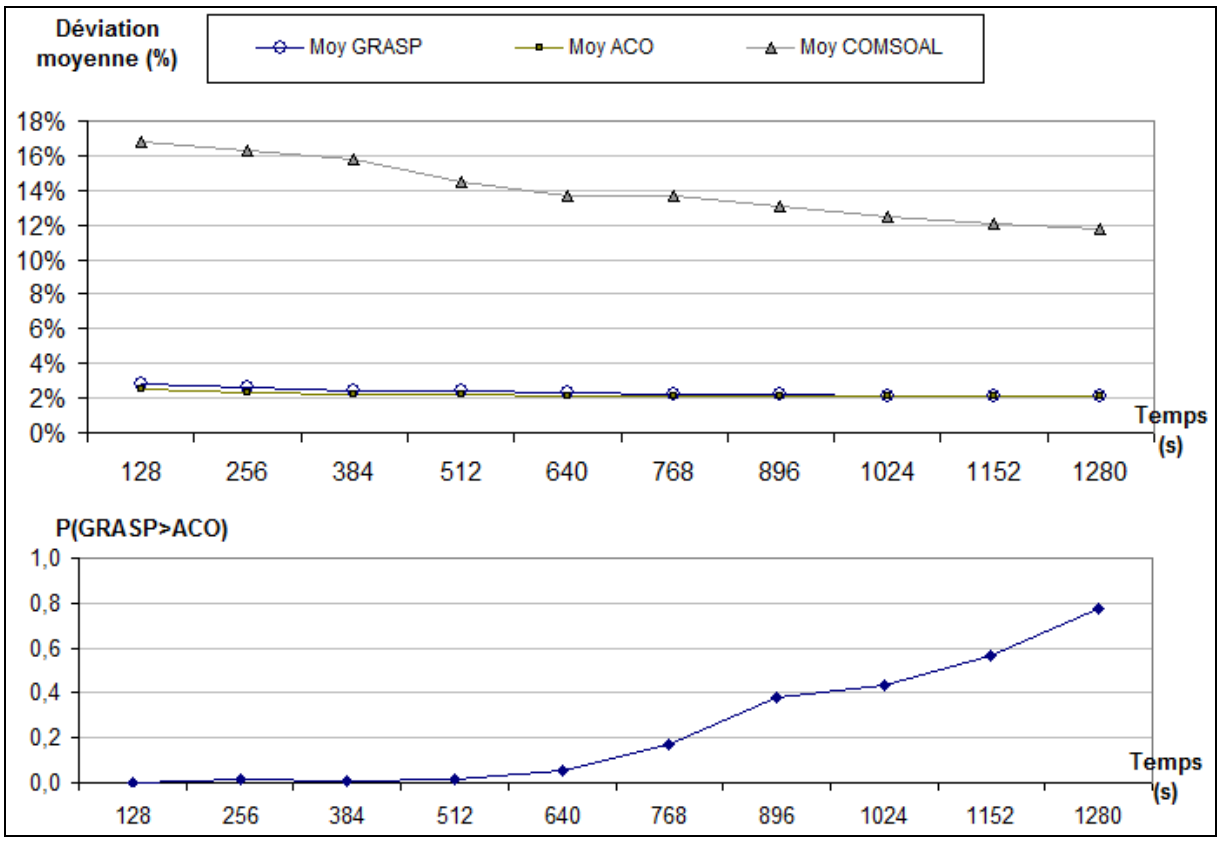

Figure 4.6 Comparaison des heuristiques itératives pour les instances de taille 80

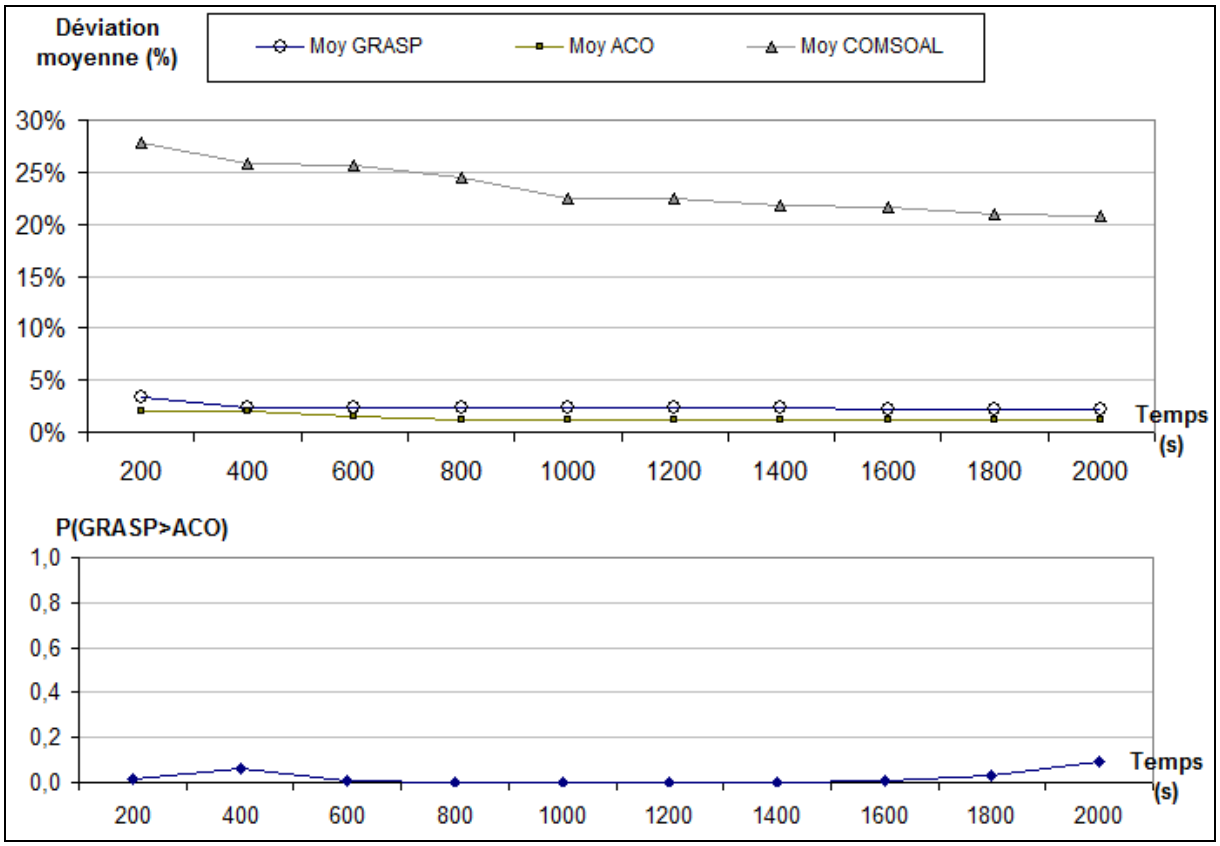

Figure 4.7 Comparaison des heuristiques itératives pour les instances de taille 100

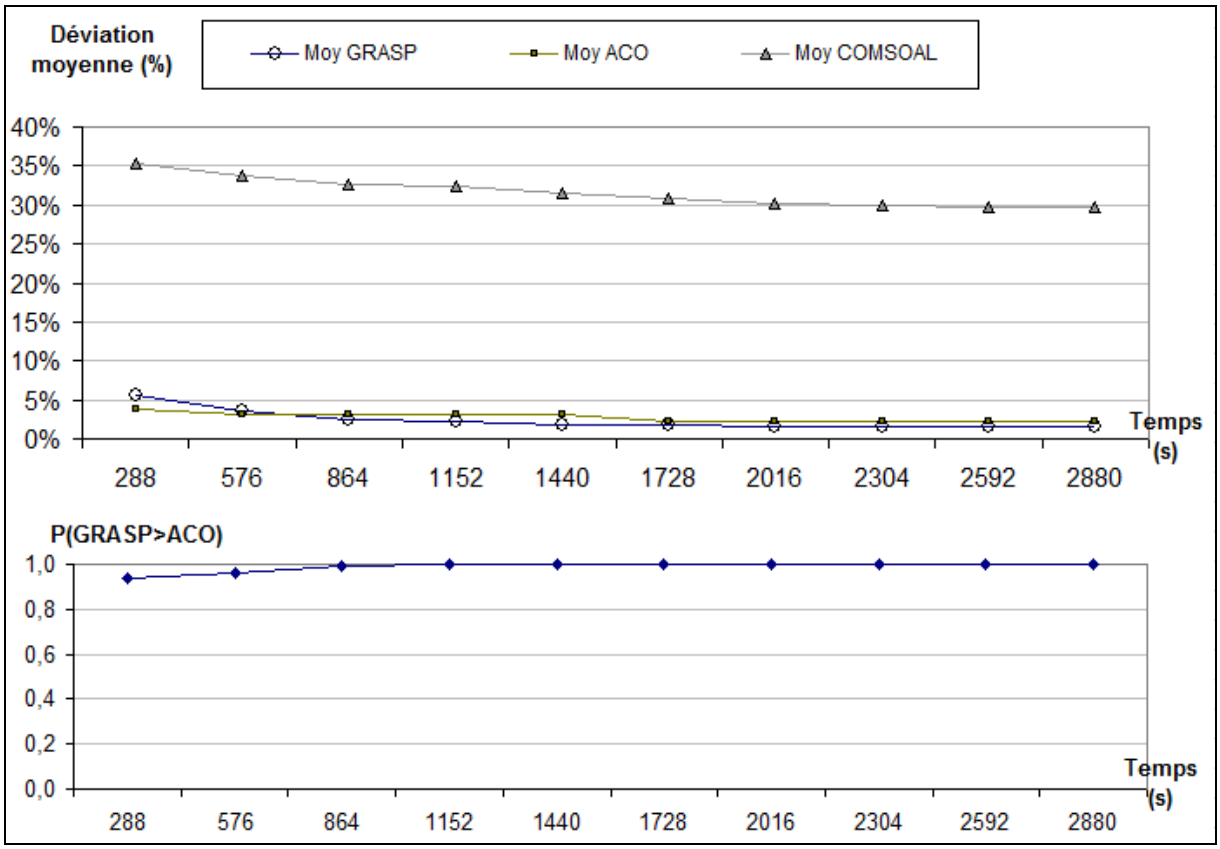

Figure 4.8 Comparaison des heuristiques itératives pour les instances de taille 120

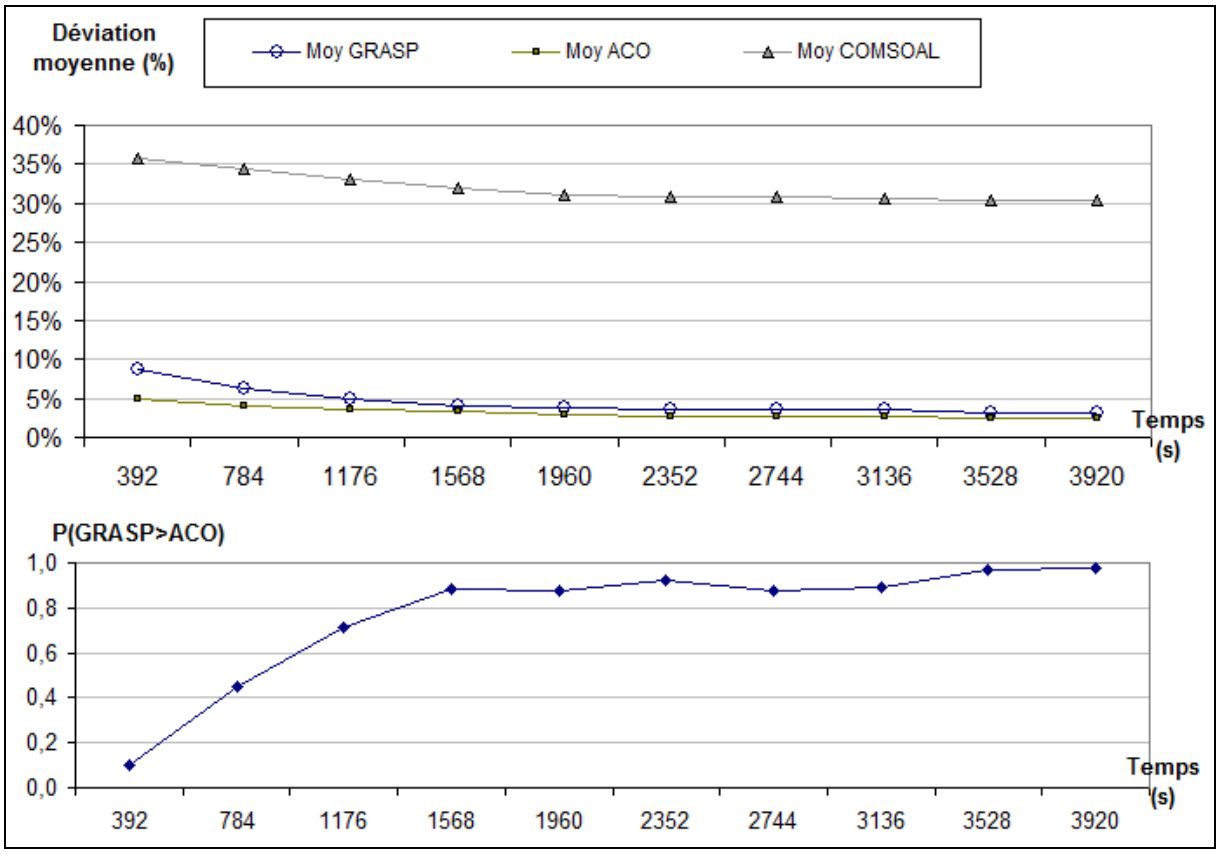

Figure 4.9 Comparaison des heuristiques itératives pour les instances de tailles 140

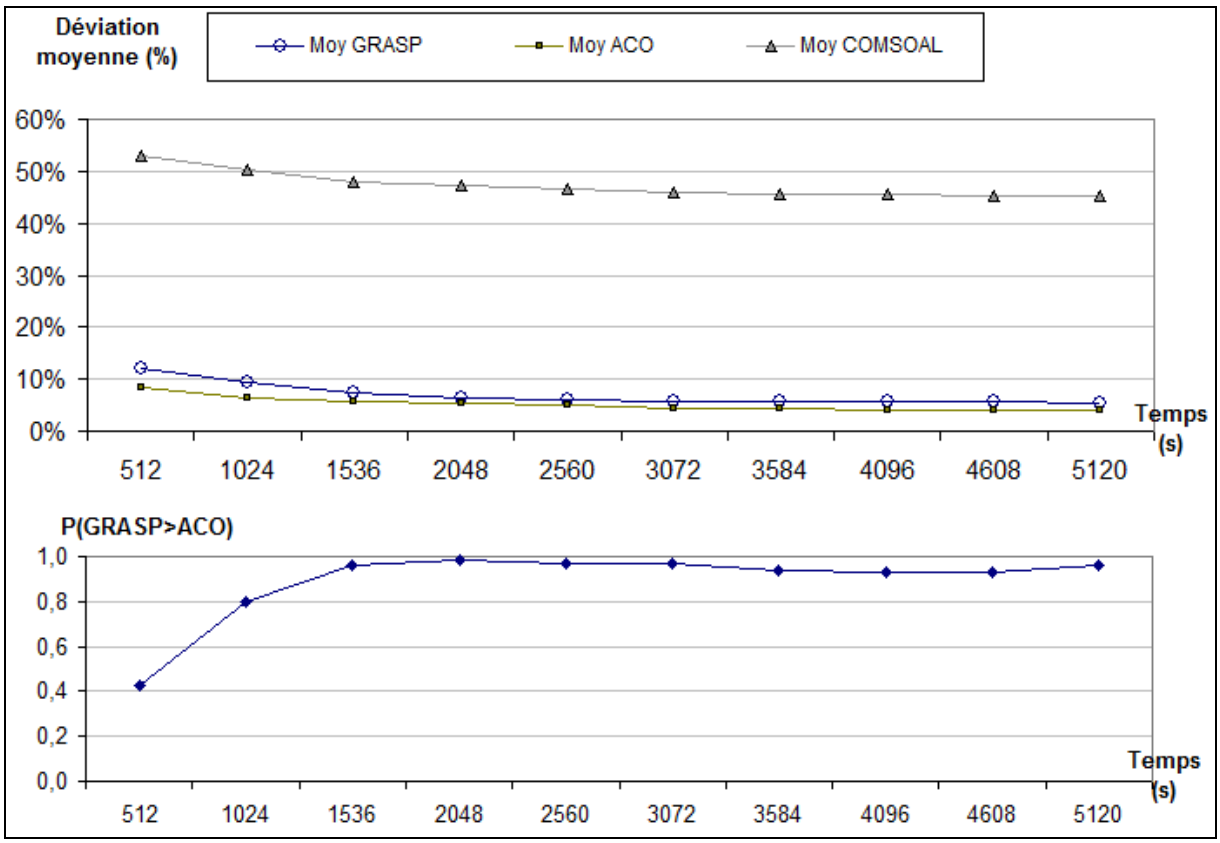

Figure 4.10 Comparaison des heuristiques itératives pour les instances de taille 160

Dans les Figures 4.3, 4.4, 4.5, 4.6, 4.7, 4.8, 4.9 et 4.10, nous présentons pour chaque famille de problème de taille différente, l'évolution de la déviation moyenne des solutions générées par GRASP, ACO et COMSOAL ainsi que l'évolution de la probabilité que GRASP soit meilleur que la méthode ACO. Nous avons effectué le test de Mann-Whitney pour comparer ACO et GRASP par rapport à COMSOAL et nous avons constaté que pour toutes les instances, l'estimation de probabilité que les méthodes GRASP ou ACO soient meilleures que COMSOAL tend vers 1.

Le coût des solutions est directement lié aux nombres de machines et de stations, ceci engendre un effet de pallier. Nous observons cet effet sur les résultats de COMSOAL qui a tendance à générer des solutions de plus mauvaise qualité pour les instances de grande taille.

Pour certaines instances, nous remarquons que l'algorithme ACO a déjà convergé et ne progresse plus tandis que GRASP parvient encore à améliorer la qualité des solutions générées. Ceci peut s'expliquer par la nature de l'algorithme utilisant les colonies de fourmi. En effet, l'utilisation de la mémoire des fourmis pour retrouver les meilleures solutions (dépôt de phéromone) peut induire un blocage sur une solution. Par contre, GRASP continue de générer des solutions variées grâce à la présence du choix aléatoire. Ainsi, même pour les instances où GRASP semble moins performant qu'ACO, il est possible qu'il s'avère efficace si le temps de calcul alloué est plus important. Afin de vérifier cela, nous nous sommes intéressés au cas des instances de taille 40 et 60. En effet, pour ces deux cas, les résultats de GRASP commencent à s'améliorer significativement au moment où le calcul est terminé. Nous avons donc relancé les deux méthodes (GRASP et ACO) sur les instances de taille 40 avec 5 fois le temps de calcul initial. Dans la Figure 4 .11, nous pouvons remarquer que les résultats de GRASP se sont nettement améliorés et ont dépassé les résultats de l'algorithme ACO que ce soit au niveau des valeurs moyennes qu'au niveau du test Mann-Witney.

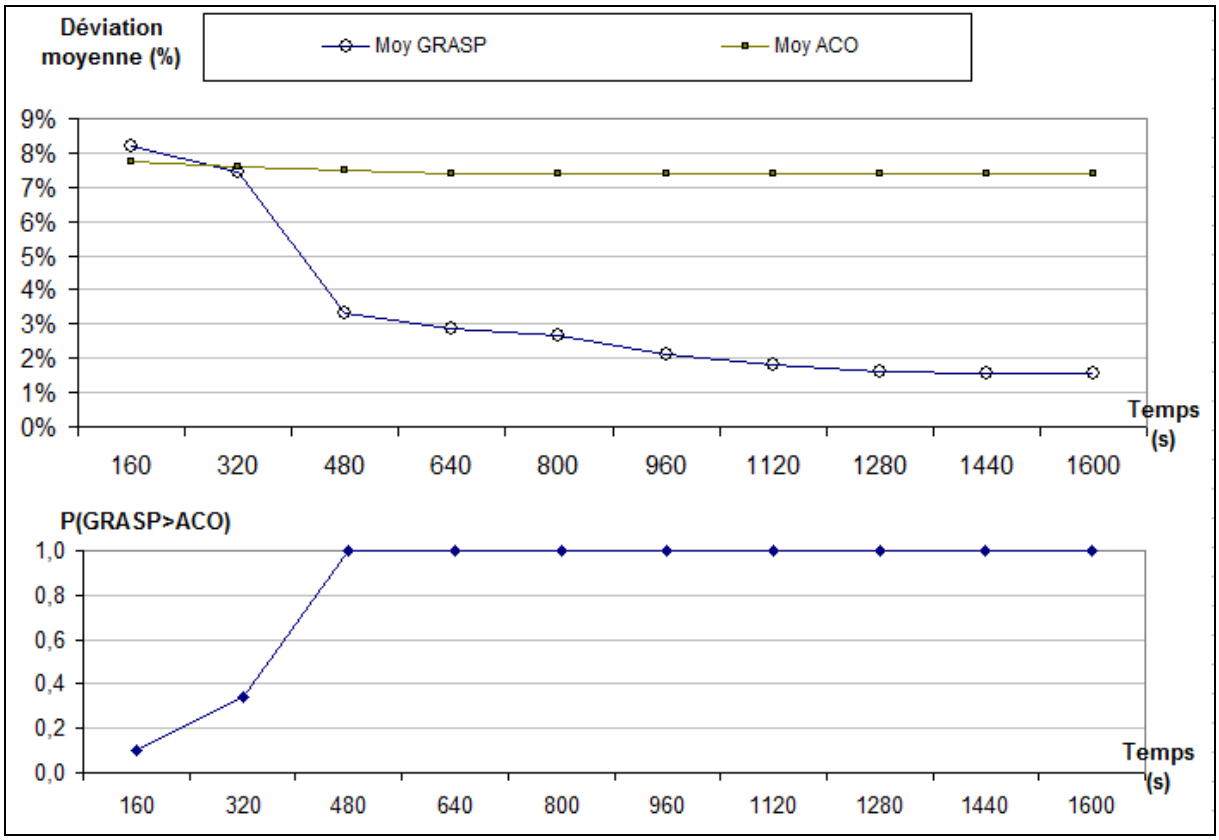

Figure 4.11 Instances de taille 40 avec temps de calcul additionnel

Pour la plupart des instances, quand la valeur moyenne des déviations de l'une des deux méthodes (GRASP ou ACO) est meilleure que celle de l'autre, la première a une bonne probabilité d'être meilleure (voir les instances 20, 40, 60, 70, 80, 100 et 120), sauf pour les instances de taille 140 et 160. Nous nous sommes interrogés sur l'effet des pires solutions sur la valeur moyenne des déviations. Pour cela, nous avons calculé la valeur médiane, le premier quartile, le troisième quartile, le minimum et le maximum des déviations des solutions générées par les deux méthodes. Nous avons illustré ces résultats dans la Figure 4.12.

Nous pouvons remarquer que même avec une valeur moyenne moins bonne des solutions générées par GRASP, à partir de 1568 secondes de calcul, la valeur médiane des solutions générées par GRASP est de meilleure qualité que celle des solutions générées par ACO, ce qui peut expliquer la valeur de la probabilité calculée. En fait, la variabilité des solutions obtenues avec GRASP s'avère plus grande, ce qui semble là encore montrer que GRASP n'a pas encore fini de converger.

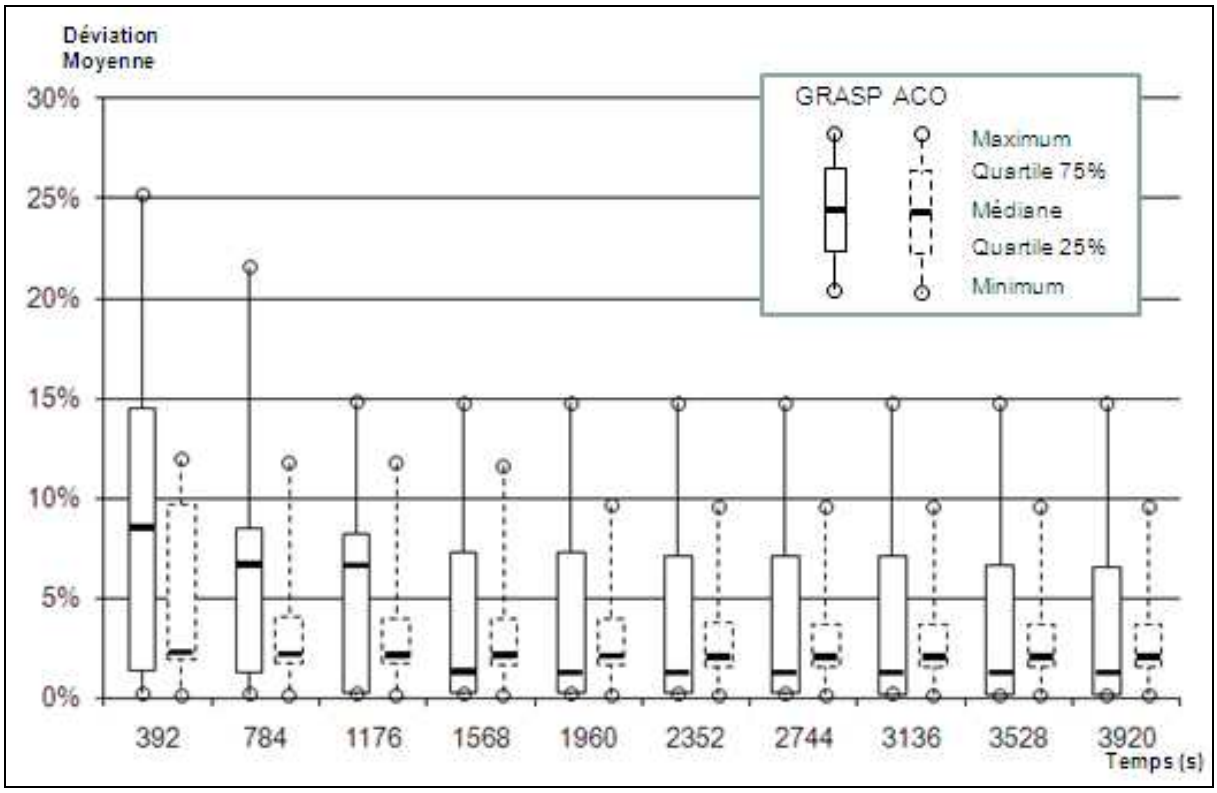

Figure 4.12 Dispersion des solutions (GRASP + ACO) pour les instances de taille 140

4.5. Conclusion

Dans ce chapitre, nous avons proposé différentes approches heuristiques pour la résolution du problème d'optimisation des lignes de transfert reconfigurables. Les approches proposées sont des méthodes constructives. Ainsi, nous avons mis en place un algorithme de construction de solution utilisant des règles de priorité pour la sélection des opérations. Trois différentes règles ont été utilisées, deux inspirés de travaux antérieurs (temps opératoire, Moodi et Young, 1965, et temps cumulé, Helgeson et Birnie, 1961) et une troisième que nous avons proposé qui permet de prendre en compte les spécificités de notre problème (temps cumulé + relations d'inclusion et de posage). Ensuite nous avons développé différents algorithmes inspirés de COMSOAL, et des métaheuristiques ACO et GRASP. L'approche basée sur GRASP a été améliorée par l'intégration d'une procédure d'ajustement du paramètre de sélection aléatoire (Reactive GRASP) ainsi qu'une procédure d'intensification basée sur l'approche de recomposition de chemin (Path-Relinking). Toutes ces méthodes ont été testées et validées sur un échantillon d'instances. GRASP et ACO se sont distinguées par rapport à COMSOAL et notre algorithme de construction avec des règles de priorité. Ces deux méthodes montrent des performances proches, même si la méthode ACO semble plus efficace

pour des temps de calcul réduits, alors que GRASP la dépasse lorsque le temps de calcul alloué est plus important.

Chapitre 5
Illustration sur un cas industriel

Dans ce chapitre, nous présentons l'utilisation de l'approche proposée dans cette thèse sur une application industrielle. Ce travail a été effectué en collaboration avec PCI SCEMM, leader sur le marché français des systèmes d'usinage. Nous considérons un cas industriel d'équilibrage de lignes pour l'usinage d'une culasse. Les caractéristiques techniques du cas sont rapportées ainsi qu'une plateforme d'optimisation mise en place en collaboration avec notre partenaire industriel. Nous exposons aussi les solutions fournies par notre algorithme GRASP et nous analysons les principaux enseignements obtenus sur ce cas industriel.

5.1. Conception de ligne de transfert en avant projet

Afin de bien situer le contexte de cette étude, nous commençons par une courte présentation de la société PCI SCEMM ainsi que de ses principaux produits et services. Nous détaillons aussi les principales étapes à suivre lors de la conception en avant projet des lignes de transfert reconfigurables.

5.1.1. Présentation de PCI-SCEMM

L'entreprise PCI-SCEMM est un leader français dans la conception et la production des systèmes d'usinage. PCI est une filiale de PSA et elle propose une gamme assez large de centres d'usinage et de machines spéciales. PCI a comme objectif de mettre à disposition de ses clients son savoir-faire dans le domaine de la conception et la fabrication des biens d'équipements pour l'usinage de pièces mécaniques en grande série, et d'accompagner ses clients pour la réduction des prix de revient des pièces fabriquées par des lignes existantes. En effet, PCI-SCEMM a développé un savoir faire dans :

- le développement d'architectures des lignes et l'analyse des flux ;
- la mise en place et la définition des processus d'usinage ;
- la conception et la fabrication de centres d'usinage à grande vitesse monobroches, bi-broches, et structures parallèles ainsi que de machines spéciales de haute productivité ;
- En plus de ces éléments, PCI prend en charge la définition de tous les moyens dans la ligne, (avec l'usinage) : l'assemblage, le lavage, la manutention et le contrôle.

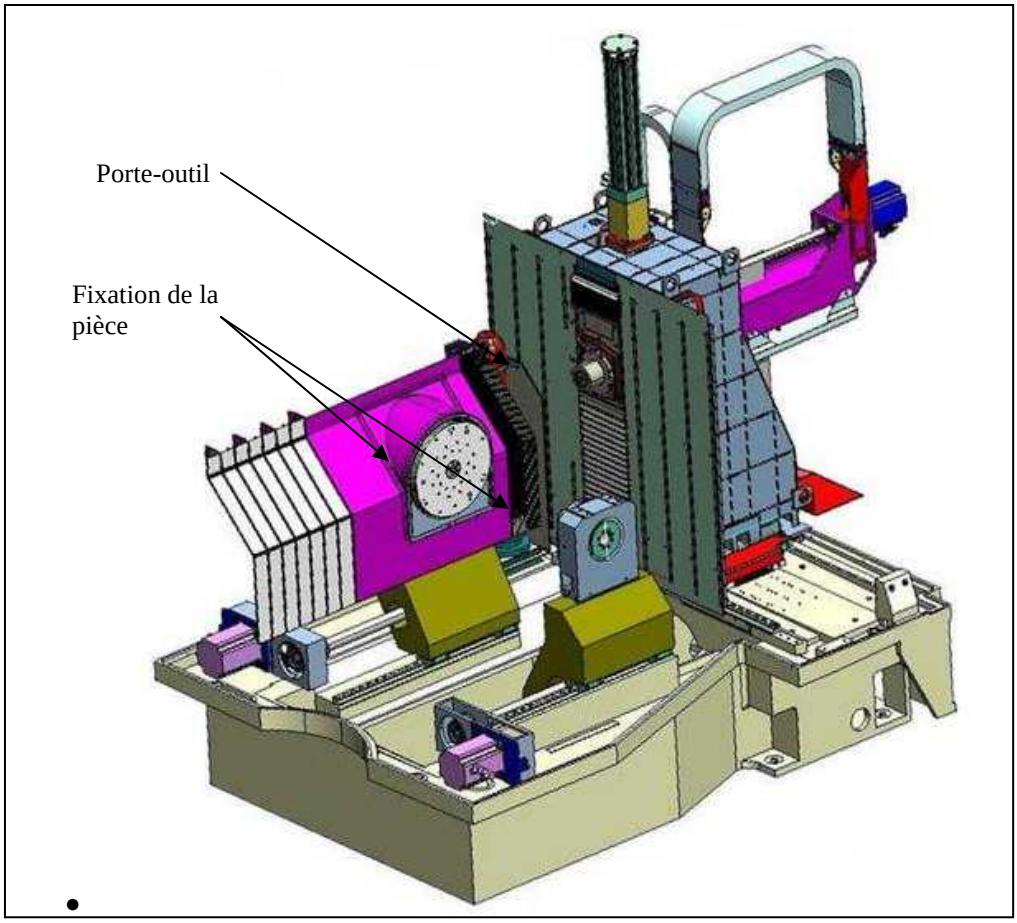

Figure 5.1 Boîtier monobroche

5.1.2. Centres d'usinage et machines spéciales

PCI propose une large gamme de centres d'usinage et de machines spéciales. Ces systèmes d'usinage se caractérisent par une haute productivité, une grande précision et l'utilisation de systèmes de commandes numériques. PCI conçoit et fabrique les produits suivants :

- Centres d'usinage horizontaux à grande vitesse monobroches (ou bi-broches) : la machine est équipée par un seul porte outil (Figure 5.1) (ou deux pour les bi-broches, Figure 5.2). Une seule opération est effectuée à la fois (deux opérations identiques sur deux pièces différentes, pour les bi-broches). Le nombre de faces accessibles par

l'outil dépend du type du centre d'usinage et de la liberté de rotation possible du plateau de fixation de la pièce.

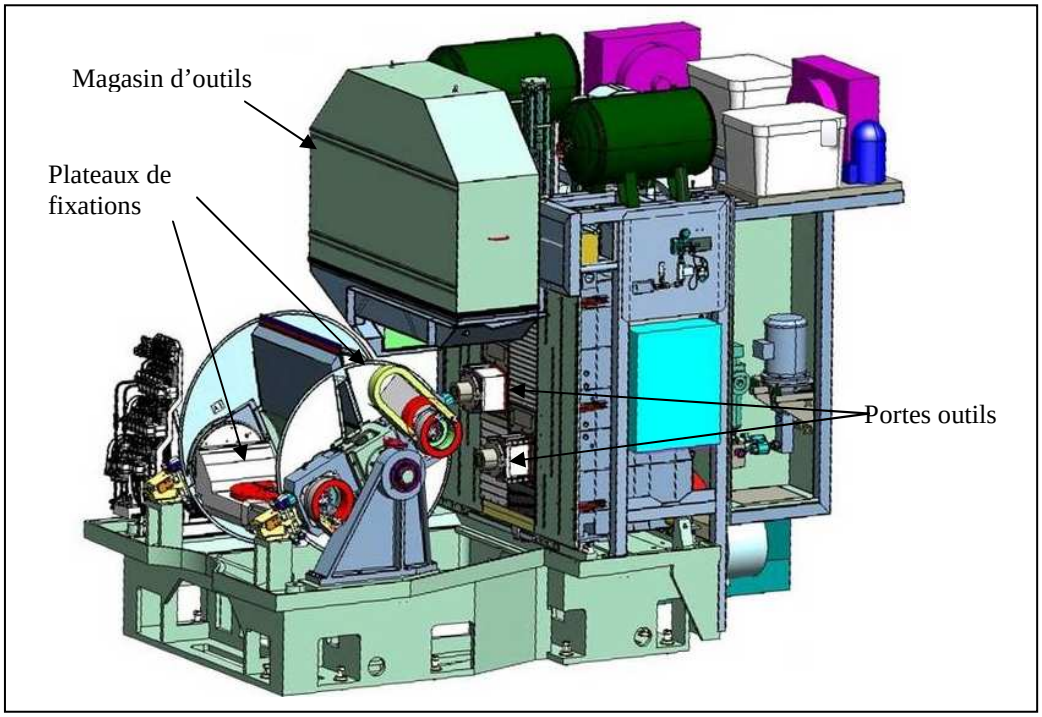

Figure 5.2 Centre d'usinage bi-broche

- Machines spéciales d'usinage : elles assurent la production en grande série de pièces mécaniques en fonte ou en aluminium. Ces machines sont adaptées au produit à usiner ainsi qu'au besoin de fabrication en termes de volume. En général, elles sont composées d'éléments standards mécaniques (boitiers, broches, tables, etc). Chaque boitier (Figure 5.3) peut ainsi comporter plusieurs broches qui effectuent des opérations d'usinage en parallèle. Une machine spéciale peut comporter plusieurs boitiers multi-broches (Figure 5.4).

- Les lignes d'usinage à base de CU et machines spéciales.

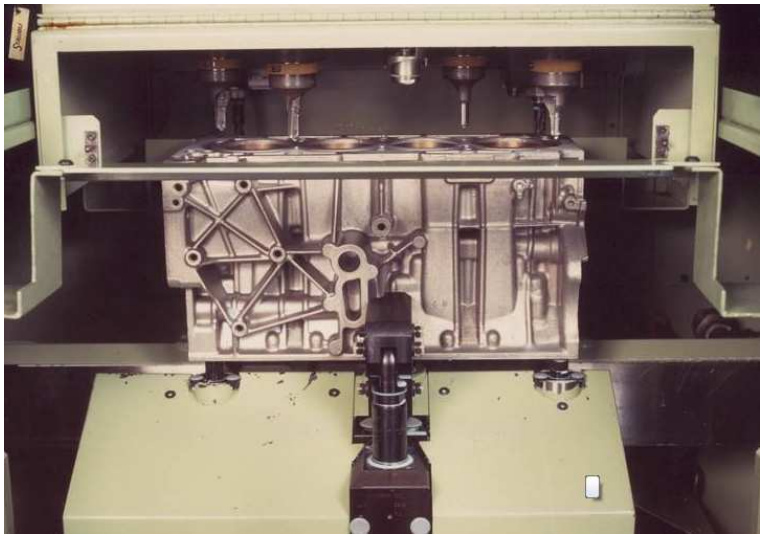

Figure 5.3 Boîtier multi-broche

Figure 5.4 Machines spéciales – à boîtiers multi-broches

5.1.3. Délais et phases lors de la conception préliminaire de ligne d'usinage

D'une façon générale, la phase de conception préliminaire d'une ligne d'usinage suit les étapes représentées dans la figure 5.5. La procédure est comme suit : un client (une compagnie industrielle) contacte l'équipementier (fabriquant de machines et de lignes d'usinage). Le client fournit les propriétés de la pièce à usiner (plans de la pièce, caractéristiques géométriques, etc.) et le volume de production exigé. À ce niveau, l'équipementier doit répondre rapidement (1-3 semaines) à la demande du client en proposant une solution préliminaire de la ligne d'usinage correspondante. Cette solution doit comporter les informations suivantes :

- L'architecture de la ligne (sauf dans le cas où le client en impose une) ;
- Les types et nombres de machines utilisées ;
- Le nombre de stations ;
- La cadence de production effective ;
- Une approximation du coût de la ligne ou du coût de production unitaire, etc.

L'acceptation par le client de la solution proposée, et par conséquent, la concrétisation de la commande, dépend de la qualité de la solution proposée dans ce laps de temps court. Une fois la commande confirmée, une période de négociation et d'échange entre l'équipementier et son client est nécessaire durant laquelle des changements au niveau de la pièce peuvent être demandés par le client, ce qui demande des modifications des processus d'usinage établis ainsi que de l'équilibrage de la ligne.

L'objectif de l'équipementier est de réduire le temps nécessaire pour la phase de conception préliminaire et de négociation de commandes et d'améliorer les performances des solutions proposées lors de cette phase afin d'optimiser au mieux les ressources utilisées et faire face à la concurrence pour gagner les marchés.

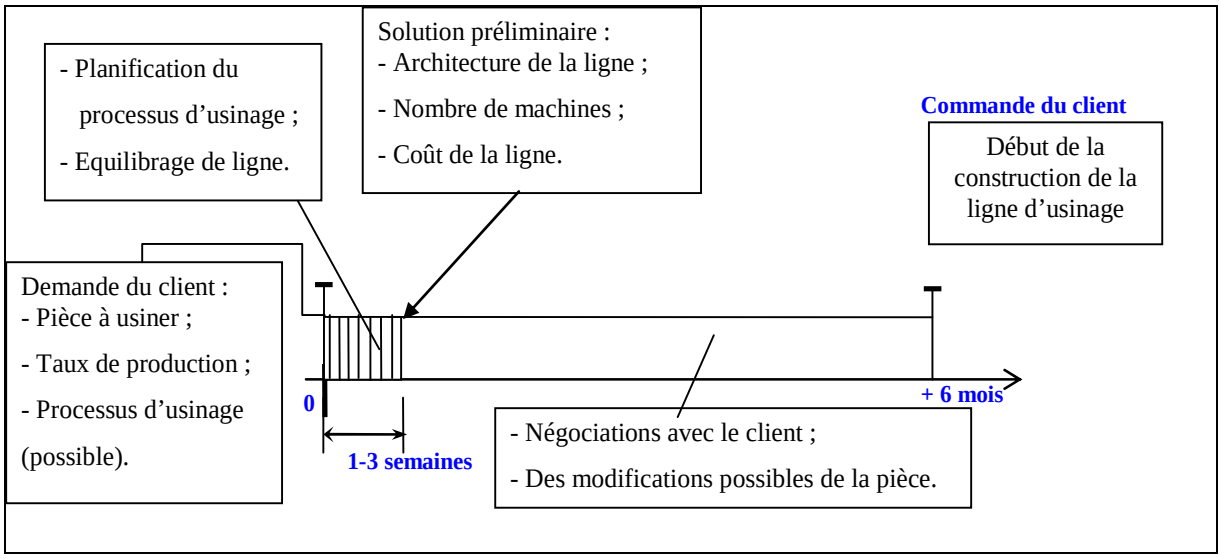

Figure 5.5 Processus de négociation: phase critique

5.2. Résolution d'un cas industriel

Dans le cadre de la collaboration entre le centre G2I de l'Ecole National des Mines de Saint-Étienne et l'entreprise PCI –SCEMM, nous avons travaillé sur un cas industriel de ligne d'usinage de culasses (Figure 5.6). Une interface de résolution développée sur Excel et faisant appel à des méthodes de résolutions algorithmiques proposées dans cette thèse a été mise en place (Figure 5.7). Cette interface contient toutes les informations du processus d'usinage (opérations d'usinage, temps opératoires, procédure de calcul des temps inter-opératoires, relations de précédence, relations d'inclusion et d'exclusion, contraintes d'accessibilité (posages)).

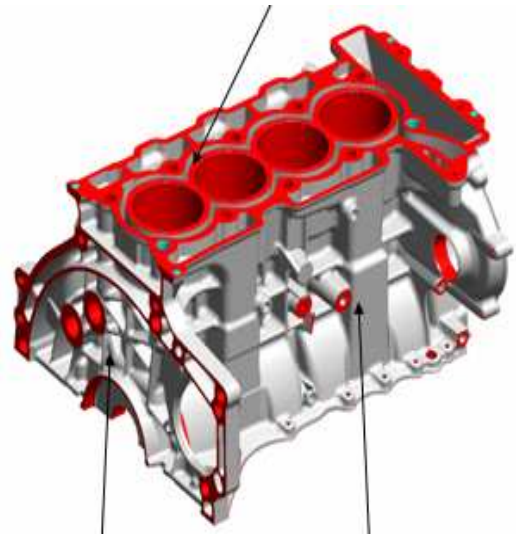

Figure 5.6 Culasse

Résolution									
N° Outil	N° Fonction	commentaire	Type usinage	Direction usinage	Type Posage (prendre en compte les précedences des	Tps VA en 1/100mn.	Précédence	Usinage sur même machine (inclusion)	Incompatibilité de fonction (machine seule)
T001	F100.01	palpage touchots	CME	100	1	12			
T002	F100.02	ebauche face comb	UFA	100	1	31	F100.01	F100.01	
T002	F200.01	ebauche face AAC sup	UFA	200	1,2	20	F100.02		
T002	F200.02	ebauche face AAC inf	UFA	200	1,2	22	F100.02		
T003	F200.03	Barettes coté AAC	UFA	200	1,2	17	x		
T003	F100.03	Barettes coté comb	UFA	100	1,2	25	x		
T004	F100.04	pointage fix culasse	UPA	100	1	4	F100.02		
T004	F100.05	pointage fix culasse	UPA	100	1	15	F100.02		
T005	F100.06	percage fix culasse (eb-DU)	UPA	100	1	10	F100.04		
T005	F100.07	percage fix culasse	UPA	100	1	24	F100.05		
T006	F100.08	Alesage finition DU	UAA	100	1	12	F100.06		
T007	F100.09	percage-alesage_locating transport	UAA	100	1	5	F100.08		
T007	F100.10	perçage fix patte	UPA	100	1,2	3	F100.08		
T007	F200.04	percage-alesage_locating MPP	UAA	200	1	5	F100.08		
T008	F200.05	lamage passage vis fix	UPA	200	1,2,5,11,15	24	06-F100.07-FC100.08		
T009	F200.06	passage clé vis fix	UPA	200	1	14	x		
T010	F200.07	Ebauche palier 4 adm et ech	UFA	200	1,2,5,11	11	F100.08		
T011	F200.08	Ebauche palier 1,2,3 adm et ech	UFA	200	1,2,5,11	11	F100.08		
T012	F200.09	Alésage eb. Puit de bougie	UAA	200	1,2,5,11,15	13	F100.08		
T013	F450.01	fraisage Face Admission	UFA	450	1,2,5,11,15	4,5	F100.08		
T013	F300.01	fraisage Face Echappement	UFA	300	1,2,5,11,15	2,5	F100.08		

Figure 5.7 Aperçu de l'interface de résolution

Par un simple clic, cette interface permet de faire appel à des méthodes de résolution grâce à une macro VBA.

5.2.1. Caractéristiques du cas d'étude

Le cas d'étude présenté ici comporte 119 opérations à exécuter. Nous considérons les contraintes de précédence, d'inclusion, d'exclusion et de posage. La spécificité de ce cas d'étude est l'utilisation de deux types de centres d'usinage :

- Centres d'usinage 4-axes : c'est un CU qui permet au porte-outil d'accéder à 4 faces différentes d'une pièce grâce à un seul axe de rotation.
- Centres d'usinage 5-axes (appellation PCI): c'est un centre d'usinage 4-axes amélioré par l'ajout d'un plateau rotatif suivant un deuxième axe. Cela permet d'avoir deux axes de rotations et donc d'accéder aux faces latérales inaccessibles. Il permet aussi l'usinage des opérations en double inclinaison. Ces machines sont plus couteuses que les CU 4-axes, ce qui rend leur utilisation limitée au seul cas de nécessité (opérations inaccessibles avec une machines 4-axes) ou de recherche de meilleure productivité.
- Quatre posages différents sont possibles pour l'exécution des différentes opérations dont un qui nécessite une machine 5-axes. Le nombre de maximal de CU par station a été fixé à 10.

5.2.2. Résultats

Nous avons donc traité ce cas en utilisant l'approche basée sur GRASP. La fonction objectif à minimiser devait en outre prendre en compte les coûts de deux types de centres d'usinage susceptibles d'être utilisés (CU de 4-axes et CU de 5-axes). Afin de mieux appréhender les différentes configurations possibles pour ce système, nous avons appliqué notre méthode d'optimisation en variant le temps de cycle imposé de la ligne de 50 à 500 unités de temps avec un pas de 25. En effet, il n'était pas exclu a priori de réaliser plusieurs lignes identiques en parallèle, chacune avec un cycle plus important, si cette dernière solution s'avérait intéressante. Dans le Tableau 5.1, nous présentons les meilleures solutions trouvées pour les différents temps de cycle. Nous précisons le temps de cycle effectif de la ligne, le nombre de CU et de stations ainsi que le nombre de CU 5-axes installées sur la ligne (les autres sont des CU de 4-axes). Nous pouvons remarquer que, parmi les solutions générées, aucune n'a un temps de cycle effectif compris entre 317 et 394 unités de temps.

Tableau 5.1 Solution par temps de cycle

coût	Temps de cycle effectif	# stations	# machines	# machines 5-axes	efficacité
12297,71	49	27	39	4	602588
12257,07	49,5	26	39	4	606725
8565,65	72	20	27	3	616727
8283,78	74	20	26	3	613000
6639,78	95	16	21	2	630779
6640,05	96	15	21	3	637445
6358	99	15	20	2	629442
5083,09	124	12	16	2	630303
4437,02	144	10	14	1	638931
3805,37	158	9	12	1	601248
3483,52	175	8	11	1	609616
3402,61	179,5	7	11	1	610768
3122,6	201	8	10	1	627643
2800,01	242	6	9	1	677602
2520,9	270	6	8	1	680643
2218,97	273	6	8	1	605779
1882,25	314	5	6	1	591027
1605,08	394	5	5	1	632402
1604,27	402	5	5	1	644917

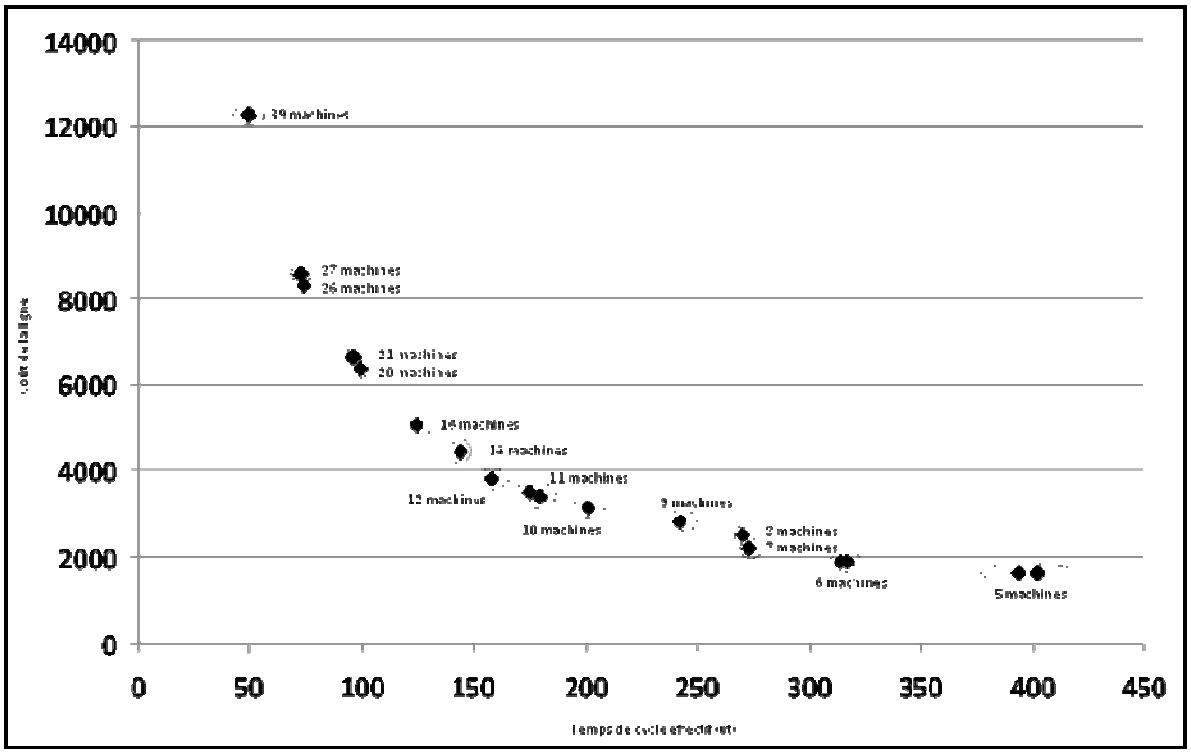

Figure 5.8 Variation du coût de la ligne en fonction du temps de cycle effectif

L'objectif de notre étude pour ce cas précis consiste à chercher un compromis entre le temps de cycle de la ligne et son coût. Dans la Figure 5.8, nous présentons l'évolution du coût des solutions générées en fonction du temps de cycle effectif. Cette courbe nous permet d'avoir une vue d'ensemble sur la combinaison coût de la ligne (ou nombre de CU)/temps de cycle. Une autre manière intéressante d'analyser les résultats, consiste à considérer la notion d'efficience utilisée dans les problèmes d'équilibrage de lignes d'assemblage. Minimiser l'efficience signifie maximiser le rendement réalisé avec le minimum de moyens. L'efficience d'une ligne est calculée par la multiplication de son coût par son temps de cycle. Ainsi, minimiser ce terme revient à minimiser le coût unitaire du produit. Pour un concepteur, cette information peut être intéressante quand il y a une possibilité de faire des lignes parallèles. Dans la Figure 5.9, nous présentons la variation de l'efficience en fonction du temps de cycle effectif. La meilleure efficience correspond à un temps de cycle égal à 314 unités de temps. Dans ce cas, il parait plus intéressant de considérer deux lignes avec un temps de cycle de 314 que la solution correspondant à une ligne unique avec un temps de cycle de 158.

Il peut aussi être utile d'observer les effets de seuil et l'impact très important que peut parfois avoir une faible variation du taux de productivité sur le coût de la ligne et inversement. Sachant que pour des études en avant projet, une marge de sécurité importante est souvent appliquée au temps de cycle souhaité pour se prémunir contre des évolutions du produit, des

imprécisions ou encore des aléas liés à la variabilité de la production ou des marchés, ce type d'information peut permettre de justifier une étude plus approfondie de configurations avec des marges plus faibles, si cela peut diminuer sensiblement le coût de la ligne.

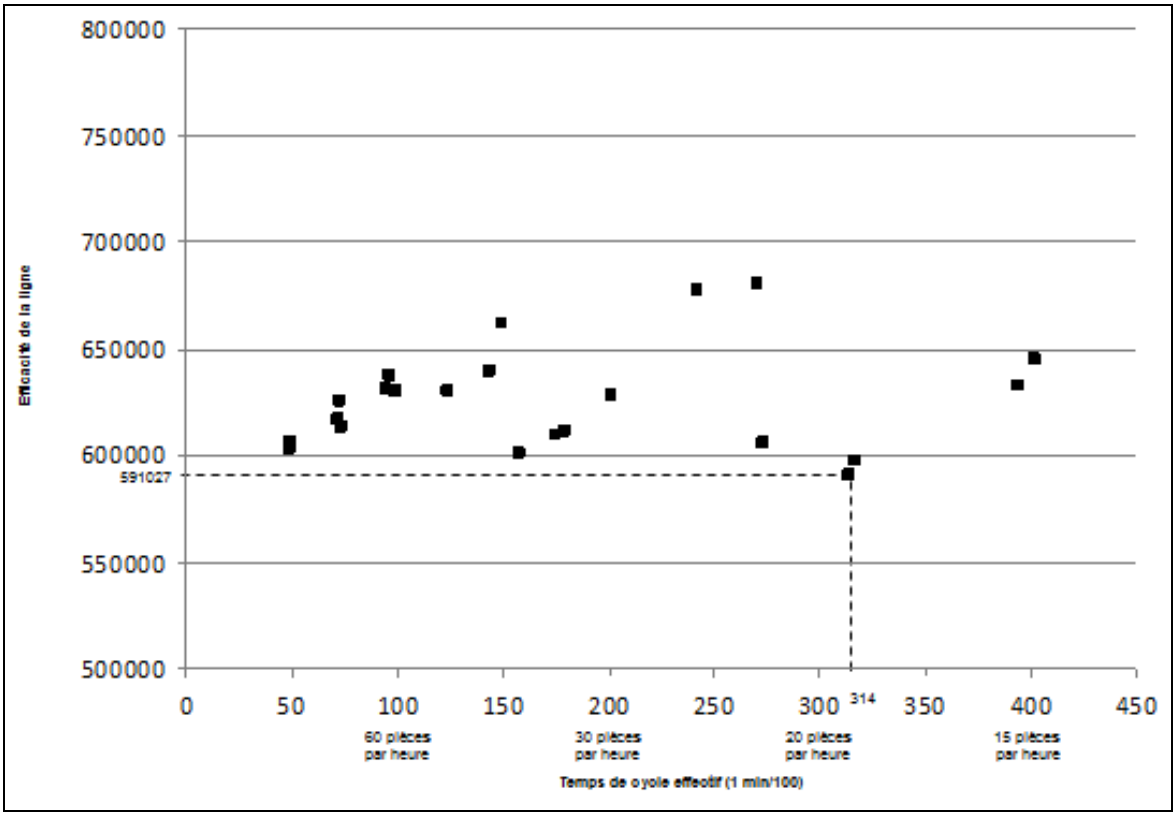

Figure 5.9 Efficience de la ligne

5.3. Conclusion

Dans ce chapitre, nous avons utilisé l'approche proposée pour la résolution d'une application industrielle d'équilibrage de ligne d'usinage. Le problème est résolu en utilisant la méthode GRASP. Ces résultats permettent d'illustrer le type d'information qu'il est possible d'obtenir avec les méthodes proposées dans ce mémoire.

Conclusion générale

Dans ce mémoire, nous avons présenté une étude sur la conception en avant projet des systèmes d'usinage reconfigurables. La particularité des systèmes étudiés réside dans l'utilisation de centres d'usinage CU mono-broches ainsi que la possibilité d'en installer plusieurs en parallèle sur la même station de travail.

Après avoir introduit les principales caractéristiques des systèmes d'usinage reconfigurables, nous avons présenté dans le deuxième chapitre, les principales méthodes de résolutions des problèmes d'optimisation combinatoires ainsi qu'un état de l'art des problèmes d'équilibrage des lignes de production. Nous nous sommes intéressés en particulier aux problèmes d'équilibrage de lignes avec stations parallèles, temps inter-opératoires, ainsi que l'équilibrage de lignes de transfert. Nous avons constaté que les problèmes d'équilibrage de lignes de transfert reconfigurables, lignes avec machines parallèles et temps inter-opératoires ne sont pas ou très peu traités dans la littérature. Le problème d'équilibrage que nous avons formulé a été formalisé grâce à des programmes linéaires en nombres mixtes. Les différentes contraintes de ces problèmes ont été modélisées, de même que les différents critères à optimiser. Une procédure de prétraitement a été développée pour faciliter la résolution de ces programmes linéaires. Des expérimentations numériques ont été effectuées afin d'évaluer les performances du modèle de base avec comme critère le coût de la ligne. La taille des instances résolues à l'optimum dans le temps alloué ne dépasse pas 18 opérations. L'approche proposée présente ainsi des limites puisque les instances industrielles peuvent comporter plus de 100 opérations. Afin de résoudre des instances de plus grande taille, nous avons donc opté pour le développement d'approches heuristiques. Nous avons proposé 3 heuristiques gloutonnes avec différentes règles de priorité. Une approche dédiée itérative avec des choix aléatoires de type COMSOAL a été développée, ainsi que 2 métaheuristiques : une basée sur

un algorithme de colonies de fourmis et la deuxième sur GRASP. Cette dernière a été améliorée par l'intégration d'une procédure d'ajustement du paramètre de sélection aléatoire (Reactive GRASP) ainsi qu'une procédure d'intensification basée sur l'approche de recomposition de chemin (Path-Relinking). Les trois approches, COMSOAL, ACO et GRASP, ont été testées sur un échantillon d'instances. Les meilleurs résultats ont été générés par GRASP et ACO, tandis que les performances de COMSOAL sont beaucoup moins bonnes surtout pour les instances de grande taille. Les résultats obtenus avec les méthodes ACO et GRASP sont proches, la première se montrant plus intéressante pour un temps de calcul réduit grâce à une convergence plus rapide, alors que la seconde est plus efficace avec un temps de calcul plus élevé. Le dernier chapitre de la thèse montre un cas industriel que nous avons traité. La méthode GRASP a été utilisée sur un problème réel d'équilibrage de lignes d'usinage à base de CU de 4 et 5 axes. Ce travail a été effectué en collaboration avec notre partenaire PCI-SCEMM. Nous avons aussi montré l'intérêt de rechercher un compromis entre le coût de la ligne et le temps de cycle.

Comme nous l'avons vu, ce travail est l'un des premiers à étudier les problèmes d'optimisation sous-jacents à la conception de lignes d'usinage reconfigurables. Il s'agit de problèmes difficiles, et il sera surement possible d'explorer d'autres pistes de résolution qui pourraient s'avérer plus efficaces. Ainsi, le modèle linéaire proposé a montré des performances nettement inférieures à celles obtenues avec le même type d'approches pour d'autres problèmes d'équilibrage de lignes de transfert déjà traités dans la littérature. Pour cela, la recherche d'une meilleure formulation du modèle linéaire pourrait être une piste intéressante. Une autre possibilité consiste à utiliser le modèle linéaire pour la résolution de sous-problèmes dans le cadre d'une résolution approchée. Ce type d'approche, appelée matheuristique (Maniezzo et al., 2010), a en effet déjà prouvé son efficacité sur un problème proche (Guschinskaya, et al., 2008). Plusieurs méthodes utilisées avec succès pour la résolution des problèmes d'équilibrage de lignes de transfert peuvent être testées et exploitées.

En outre, si ce travail a permis de développer des méthodes capables d'appréhender efficacement les contraintes propres aux lignes reconfigurables, le caractère intrinsèquement multicritère de ce problème, qui a notamment été mis en évidence sur le cas industriel, justifierait la mise en œuvre de méthodes d'optimisation multi-objectif. Dans ce cadre, il paraît important de pouvoir disposer, en plus des critères de coût et de taux de production

considérés dans ce mémoire, d'un ou plusieurs indicateurs quantitatifs pour évaluer le niveau de reconfigurabilité d'une ligne d'usinage.

Pour finir, le concept de lignes reconfigurables a été proposé pour définir des caractéristiques technologiques et des architectures de lignes permettant de répondre aux variations et aux incertitudes de plus en plus grandes sur l'évolution de la demande (volume de production, caractéristiques du produit, …). Cette problématique de la prise en compte d'incertitudes a par ailleurs donné lieu ces dernières années à un nombre important de travaux sur des notions telles que la robustesse ou la stabilité des solutions obtenues (voir par exemple Billaut et al., 2005 et Sotskov et al., 2006 sur des problèmes proches). Ces approches (reconfigurabilité et robustesse d'une solution) paraissent complémentaires et leur comparaison pourrait susciter de nouvelles pistes de recherche pour répondre à ce problème crucial.

Bibliographie

Alvarez-Valdes, R., Crespo, E., Tamarit, J.M. et Villa, F., 2008. GRASP and path relinking for project scheduling under partially renewable resources. *European Journal of Operational Research*, 189, p. 1153-1170.

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, p. 255-264.

Amen, M., 2002. Heuristic methods for cost oriented assembly line balancing: A survey. *International Journal of Production Economics*, 68, p. 1-14.

Andrés, C., C. Miralles et R. Pastor, 2008. Balancing and scheduling tasks in assembly lines with sequence-dependent setup times. *European Journal of Operational Research*, 187, p. 1212–1223.

Arcus, A.L., 1965. COMSOAL: a computer method of sequencing operations for assembly lines. *International Journal of Production Research*, 4(4), p. 259-277.

Askin, R.G. et C.R. Standridge, 1993. Modeling and analysis of manufacturing systems. John Wiley &Sons, Inc.

Askin, R.G. et M. Zhou, 1997. A parallel station heuristic for the mixed-model production line balancing problem. *International Journal of Production Research*, 35(11), p. 3095-3105.

Awad, M., Kusela, J. et Ziegler, J., 1996. Object-Oriented Technology for Real-Time Systems. Prentice Hall, NJ.

Baptiste, P., Le Pape, C. et Nuijten, W., 2003. Constraint-based scheduled Applying Constraint Programming to Scheduling Problems. *Kluwer Academic Publisher*, London.

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

Bautista J., Pereira J., 2007. Ant algorithms for a time and space constrained assembly line balancing problem. *European Journal of Operational Research*, 177(3), p. 2016-2032.

Baybars, I., 1986. A survey of exact algorithms for the simple assembly line balancing problem, *Management Science*, 32(8), p. 909-932.

Belhoste, J.F., S., Benoit, S., Chassagne, P., Mioche, 2004. Autour de l'industrie : Histoire et patrimoine. *La Documentation Française*, 642 pages.

Bellman, R., 1957. Dynamic Programming. *Princeton University Press*, Princeton.

Bellman, R. et S.E. Dreyfus, 1962. Applied Dynamic Programming. Princeton University Press, Princeton.

Belmokhtar, S., 2006. Lignes d'usinage avec équipements standards: modélisation, configuration et optimisation. Thèse de doctorat, Ecole des Mines de Saint-Etienne, France.

Belmokhtar, S., Dolgui, A., Guschinsky, N. et Levin, G., 2006. An integer programming model for logical layout design of modular machining lines. *Computers and Industrial Engineerin*g, 51(3), p. 502-518.

Berrichi A., Yalaoui F., Amodeo L. et Mezghiche M., 2010. Bi-Objective Ant Colony Optimization approach to optimize production and maintenance scheduling. *Computers & Operations Research*, 37(9), p. 1584-1596.

Bhattachajee, T.K. et Sahu, S., 1990. Complexity of Single Model Assembly Line Balancing Problems. *Engineering Costs and Production Economics*, 18, p. 203-214.

Bigras, L.P., Gamche, M., Savard, G., 2008. The time-dependent traveling salesman problem and single machine scheduling problems with sequence dependent setup times. *Discrete Optimization*, 5, p. 685-699.

Billaut, J.C., Moukrim A., Sanlaville E., 2005. Flexibilité et robustesse en ordonnancement. Lavoisier, 344 pages.

Boctor, F.F., 1995. A multiple-rule heuristic for assembly line balancing. Journal of *Operational Research Society*, 46, p. 62-69.

Boudia, M., Louly, M.A.O. et Prins, C., 2007. A reactive GRASP and path relinking for a combined production-distribution problem. *Computer and Operations Research*, 34, p. 3402-3419.

Bowman, E.H., 1960. Assembly-Line Balancing by Linear Programming. *Operations Research*, 8(3), p. 385-389.

Boysen, N., Fliedner, M. et Scholl, A., 2007. A classification of assembly line balancing problems. *European Journal of Operational Research*, 183(2), p. 674-693.

Boysen, N., Fliedner, M. et Scholl, A., 2008. Assembly line balancing: which model to use when? International Journal of Production Economics, 111, p. 509–528.

Bukchin, J. et Rubinovitz, A., 2002. A weighted approach for assembly line design with station paralleling and equipment selection. *IIE Transactions*, 35, p. 73-85.

Bukchin, J. et Tsur, M., 2000. Design of flexible assembly line to minimize equipment cost. *IIE Transactions*, 32, p. 585-598.

Buxey, G.M., 1974. Assembly line balancing with multiple stations. *Management Science*, 20, p. 1010-1021.

Carlier, J., 1982. The one-machine sequencing problem. *European Journal of Operational Research*, 11, p. 42-47.

Carlier, J. et Chretienne, P., 1988. Problème d'ordonnancement; modélisation / complexité / algorithmes. Editions Masson.

Carlier, J. et Pinson, E., 1989. A Branch and Bound Method for Solving the Job Shop Problem. *Management Science*, 164-176.

Colmerauer, A., 2002. Cours de DEA sur la programmation logique. http://www.lim.univ-mrs.fr/.

Cottle, R.W., 2003. The Basic George B. Dantzig. Richard W. Cottle Ed., Stanford University Press, Stanford, California.

Dantzig, G.B., 1959. Note on Solving Linear Programs in integers. *Naval Research Logistics Quarterly*, 6, p. 75-76

Dar-El Mansoor, E.M., 1964. Assembly Line Balancing: an improvement on the ranked positional technique. *Journal of Industrial Engineering*, 15(2), p. 73-77.

Dashchenko, A.I. (Ed.), 2003. Manufacturing Technologies for Machines of the Future: 21st Century Technologies. Springer Verlag, Berlin- Heidelber.

Dashchenko, A.I. (Ed.), 2006. Reconfigurable Manufacturing Systems and Transformable Factories. Springer Verlag, Berlin- Heidelber.

Delmaire, H., Diaz, J.A., Fernandez, E. et Ortega, M., 1999. Reactive GRASP and tabu search based heuristics for the single source capacitated plant location problem. *Information Systems and Operational Research,* 37(3), p. 194-225.

Delorme, X., Gandibleux, X. et Rodriguez, J., 2004. GRASP for set packing problems. *European Journal of Operational Research,* 153, p. 564-580.

Demeulemeester, E. et Herroelen, W., 1992. A branch-and-bound procedure for the multiple resource-constrained project scheduling problem. *Management Science*, 38, p. 1803-1818.

Desrochers, M. et Laporte, G., 1991. Improvement and extensions to the Miller-Tucker-Zemlin subtour elimination constraints. *Operations Research Letters*, 10, p. 27-36.

Dietrich, R., 1992. Precis De Methodes D'usinage - Méthodologie, Production Et Normalisation, 5ème Édition, Broché.

Dolgui, A. (Ed.), 2006. Feature Cluster on the Balancing of Assembly and Transfer Lines. *European Journal of Operational Research,* 168(3), p. 663 – 951.

Dolgui, A., Finel, B., Guschinsky, N., Levin G. et Vernadat, F., 2005. An heuristic approach for transfer lines balancing. *Journal of Intelligent Manufacturing*, 16(2), p. 159-171.

Dolgui, A., Finel, B., Guschinsky, N., Levin G. et Vernadat, F., 2006a. MIP Approach to Balancing Transfer Lines with Blocks of Parallel Operations. *IIE Transactions*, 38, p. 869-882.

Dolgui, A., Guschinskaya, O., Guschinsky N. et Levin, G., 2007. Decision Making and Support Tools for Design of Machining Systems. Chapter in: Encyclopedia of Decision Making and Decision Support Technologies, F. Adam and P. Humphreys (Eds.), Idea Group Inc.

Dolgui, A., Guschinskaya, O., Guschinsky, N. et Levin, G., 2009. Decision Making and Support Tools for Design of Machining Systems, Chapter in: *Strategic Information Systems: Concepts, Methodologies, Tools, and Applications*, M. Gordon Hunter (Ed.), Idea Group Inc., (ISBN-13: 978-1-60566-677-8), 2009, vol. 2, *Development and Design Methodologies* (Reprint of the chapter from *Encyclopedia of Decision Making and Decision Support Technologies* made by the Publishing House).

Dolgui, A., Guschinsky N. et Levin, G., 1999. On Problem of Optimal Design of Transfer Lines with Parallel and Sequential Operations. *Proceedings of the 7th IEEE International Conference on Emerging Technologies and Factory Automation* (ETFA'99). October 18-21, 1999, Barcelona, Spain. J.M. Fuertes (Ed.), IEEE, 1, p. 329-334.

Dolgui, A., Guschinsky, N. et Levin, G., 2006b. A Special case of transfer lines balancing by graph approach. *European Journal of Operational Research,* 168(3), p. 732-746.

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

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

Dolgui, A. et Pashkevich, A., 2001. Manufacturing Systems Design. Université de Technologie de Troyes, 304 pages.

Dolgui, A. et Proth, J.M., 2006. Les systèmes de production modernes. Hermes-Science.

Dolgui, A. et Proth. J.M., 2010. Supply Chain Engineering: Useful Methods and Techniques. Springer, 2010, (ISBN: 978-1-84996-016-8), 539 pages. (à paraître).

Dorigo, M. 1992. Optimization, learning and natural algorithms (in Italian). Ph,D, Thesis, Dipartimento di Elettronica, Politecnico di Milano.

Dorigo, M. et Di Caro, G., 1999. Ant colony optimization: a new meta-heuristic. In: Angeline Peter J, Michalewicz Zbyszek, Schoenauer Marc, Yao Xin, Zalzala Ali, editors, Proceeding of the congress on evolutionary computation, IEEE press, 2, p. 1470-1477,

Dorigo M., Gambardella, L.M., 1997. Ant colonies for the travelling salesman problem, BioSystems, 43, p. 73-81.

Erel, E. et Sarin, S.C.,1998. A survey of the assembly line balancing procedures. *Production Planning and Control*, 9(5), p. 414–434.

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

Essafi, M., Delorme, X., Dolgui, A., 2010b. Balancing machining lines: a two-phase heuristic, *Studies in Informatics and Control*, 19(3), 243-252.

Essafi, M., Delorme, X., Dolgui, A., 2010c. Balancing lines with CNC machines: a multi-start Ant based heuristic. *CIRP Journal of Manufacturing Science and Technology*, 2, p. 176-182.

Feo, W.E. et Resende, M.G.C., 1989. A probabilistic heuristic for a computationally difficult set covering problem, *Operations Research Letter*, 8, p. 67-71.

Finel, B., 2004. Structuration de lignes d'usinage : méthodes exactes et heuristiques. Thèse de doctorat, Université de Metz, France.

Finel, B., Dolgui A. et Vernadat, F., 2008. A Random Search and Backtracking Procedure for Transfer Line Balancing. *International Journal of Computer Integrated Manufacturing*, 21(4), p. 376-387.

Gadidov, R. et Wilhelm, W., 2000. A cutting plane approach for the single-product assembly system design problem. *International Journal of operational Research*, 35(3-4), 467-470.

Garey, M.R. et Johnson, D.S., 1979. Computers and intractability: A guide to the theory of NP-Completeness. Bell Laboratories Murray Hill, New Jersey.

Ghosh, S. et Gagnon, R.J., 1989. A comprehensive literature review and analysis of the design, balancing and scheduling of assembly line systems. *International Journal of Production Research*, 27, p. 637-670.

Glover, F., 1996. Tabu search and adaptive memory programming - Advances, applications and challenges. In: Barr, R.S., Helgason, R.V. et Kennington, J.L., eds,, *Interfaces in Computer Science and Operations Research*, Kluer, p. 1-75.

Glover, F., Laguna, M. et Marti, R., 2000. Fundamentals of scatter search and path relinking. *Control and Cybernetics*, 39, p. 653-684.

Groover, M.P., 1987. Automation, Production Systems and Computer Integrated Manufacturing, Prentice Hall, Eaglewood Cliffs, New Jersey.

Guschinskaya, O., 2007. Outils d'aide à la décision pour la conception en avant-projet des systèmes d'usinage à boîtiers multi-broches. Thèse de doctorat, Ecole des Mines de Saint Etienne, France.

Guschinskaya, O., Dolgui, A., Guschinsky, N. et Levin, G., 2008. A heuristic multi-start decomposition approach for optimal design of serial machining lines. *European Journal of Operational Research*, 189(3), p. 902-913.

Hanafi S., Lazic, J., Mladenovic, N. et Wilbaut, C., 2009. Variable Neighborhood decomposition Search with Bounding for Multidimensional Knapsack Problem. *Proceeding of 13th IFAC Symposium on Information Control Problems in Manufacturing*, Moscow, Russia, 3-5 June 2009.

Helgeson, W.P. et Birnie, D.P., 1961. Assembly line balancing using the ranked positional weight technique. *Journal of Industrial Engineering*, 12, p. 394-398.

Hitomi, K., 1996. Manufacturing system engineering. Taylor&Francis.

Jackson, J.R., 1956. A computing procedure for a line balancing problem. *Management Science*, 2, p. 261-271.

Jones, D.R. et Beltramo, M.A., 1991. Solving partitioning problems with genetic algorithms. *Proceeding of IGGA91*, San Diego, USA, p. 442-449.

Khachian, L.G., 1979. A Polynomial Algorithm in Linear Programming. *Dokl. Akad. Nauk SSSR* 244, p. 1093-1096, 1979. English translation in *Soviet Math. Dokl.* 20, p. 191-194, 1979.

Karmarkar, N., 1984. A New Polynomial-Time Algorithm for Linear Programming. *Combinatorica* 4, p. 373-395.

Kim, J.Y., Kim, Y. et Kim, Y.K., 2001. An Endosymbiotic evolutionary algorithm for optimization. *Applied Intelligence*, 15(2), p. 117-130.

Kirkpatrick, S., Gelatt, Jr. C. D. et Vecchi, M. P., 1983. Optimization by Simulated Annealing. *Science*, 220(4598), p. 671-680.

Koren, Y., Heisel, U., Javane, F., Moriwaki, T., Pritchow, G., Van Brussel, H. et Ulsoy, A.G., 1999. Reconfigurable manufacturing systems. *CIRP Annals*, 48(2), p. 527-598.

Kusiak, A., 1986. Modelling and Design of Flexible Manufacturing Systems, Elsevier, Amsterdam-Oxford-New York-Tokyo.

Laguna, M. et Marti, R., 1999. GRASP and Path Relinking for 2-Layer Straight Line Crossing Minimization. *Informs Journal on Computing*, 11(1), p. 44-52.

Lapierre, S.D., Ruiz, A. et Soriano, A., 2006. Balancing assembly lines with tabu search. *European Journal of Operational Research*, 168(3), p. 826-837.

Maniezzo, V., Stutzle, T., Voss, S. (eds), 2010. Matheuristics: Hybridizing Metaheuristics and Mathematical Programming. *Annals of Information Systems*, (ISBN: 978-1-4419-1305-0), Springer, 10.

Masood, S., 2006. Line balancing and simulation of an automated production transfer line. *Assembly Automation*, 26(1), p. 69-74.

Matta, A. et Semeraro, Q., 2005. Design of Advanced Manufacturing Systems. Springer, 2005.

McMullen, P.R. et Frazier, G.V., 1998. Using simulated annealing to solve a multiobjective assembly line balancing problem with parallel workstations. *International Journal of Production Research*, 39(10), p. 2717-2741.

Moodi, C.L. et Young, H.H., 1965. A heuristic method for assembly line balancing for assumption of constant or variable elements time. *Journal of Industrial Engineering*, 16, p. 23-29.

Nemhauser, G. et Wolsey, L.A., 1998. Integer and Combinatorial Optimization. Wiley-Interscience Publication.

Nof, S.Y., Wilhelm, W.E. et Warnecke, H.J., 1997. Industrial Assembly, Chapman & Hall.

Osman, I.H. et Laporte, G., 1996. Metaheuristics : a bibliography. *Annals of Operations Research*, 63, p. 513-623.

Paschos V., 2005. Optimisation combinatoire 1, concepts fondamentaux. Hermes Sciences Publishing.

Patterson, J.H. et Albracht, J.J. 1975. Assembly line balancing: 0-1 programming with Fibonacci search. *Operations Research*, 23, p. 166-174.

Pinto, P.A., Dannenbring, D.G. et Khumawala, B.M., 1975. A branch and bound algorithm for assembly line balancing with paralleling. *International Journal of Production Research*. 13(2), p. 183-196.

Pinto, P.A., Dannenbring, D.G. et Khumawala, B.M., 1981. Branch and bound and heuristic procedures for assembly line balancing with paralleling of stations. *International Journal of Production Research*, 19(5), p. 565-576.

Ponnambalam, S.G., Aravindan P. et Mogileeswar Naidu, G., 1999. A comparative evaluation of assembly line balancing heuristics. *International Journal of Advanced Manufacturing Technology*, 15(8), p. 577-586.

Prais, M. et Ribeiro, C.C., 2000. Reactive GRASP: an application to a matrix de composition problem in TDMA traffic assignment. *Journal on Computing*, 12(3), p. 164-176.

Pritschow, G. et Muller, J., 1997. Object Oriented Modelling of Numerical Control Modules, *Production engineering*, 4, p. 83-86.

Rekiek, B., Dolgui, A. Delchambre A. et Bratcu, A., 2002. State of the art of assembly lines design optimization. *Annual Reviews in Control*, 26(2), p. 163-174.

Rekiek, B., De Lit, P. et Delchambre, A., 2000. Designing mixed-product assembly lines. *IEEE Transactions on Robotics and Automation*, 16(3), p. 268-280.

Resende, M.G.C., Marti, R., Gallego, M. et Duarte, A., 2010. GRASP and path relinking for the max-min diversity problem. *Computer and Operations Research*, 37, p. 498-508.

Resende, M.G.C. et Ribeiro, C.C., 2006. Greedy Randomized Adaptive Search Procedures: advances and applications. In: Potvin J.Y., and Gendriau, M., eds, *International Series in Operations Research & Management Science*, handbook of meta-heuristics, 57, p. 219-249.

Rubinovitz; J. et Levitin, G., 1995. Genetic algorithm for assembly line balancing. *International Journal of Production Economics*, 41(1-3), p. 343-354.

Sabuncuoglu I., Erel E. et Alp A., 2009. Ant colony optimization for the single model U-type assembly line balancing problem. *International Journal of Production Economics*, 120, p. 287-300.

Salvendy, G., (ed.), 2001. Handbook of Industrial Engineering, Technologiy and Operations Management.

Salveson, M.E., 1955. The assembly line balancing problem. *Journal of Industrial Engineering*, 6(4), p. 18-25.

Scholl, A., 1999. Balancing and sequencing of assembly lines. Heidelberg, Physica.

Scholl, A. et Becker, C., 2006. State-of-the-art exact and heuristic solution procedures for simple assembly line balancing. *European Journal of Operational Research*, 168, p. 666-693.

Scholl, A., Boysen N. et Fliedner, M., 2008. The sequence-dependent assembly line balancing problem. Operational Research Spectrum, 30(3), p. 579-609.

Scholl, A. et Klein, R., 1999. Balancing assembly line effectively – A computational comparison. *European Journal of Operational Research*, 114, p. 50-58.

Seriano, P. et Gendreau, M., 1997. Fondements et applications des méthodes de recherche avec tabou. *RAIRO Operations Research*, 31, p. 133-159.

Sotskov, Y., Dolgui, A. et Portmann, M.C., 2006. Stability analysis of an optimal balance for an assembly line with fixed cycle time. European Journal of Operational Research, 168, p. 783-797.

Spicer P., Koren Y., Shpitalni M. et Yip-Hoi D., 2002. Design Principles for Machining System Configurations. *CIRP Annals-Manufacturing Technology*, 51(1), p.275-280.

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

Suresh, G. et Sahu, S., 1994. Stochastic assembly line balancing using simulated annealing. *International Journal of Production Research*, 32, p. 1801-1810.

Szadkowski, J., 1997. Critical path concept for multi-tool cutting processes optimization. In: P. Kopacek, *Manufacturing Systems Modeling, Management and Control: Proceedings of the IFAC Workshop*, Vienna, Austria. Elsevier, p. 393-398.

Talbot, F.B., Patterson J.H. et Gehrlein, W., 1986. A comparative evaluation of heuristic line balancing techniques. *Management Science*, 32, p. 430-454.

Vilarinho, P.M. et Simaria, A.S., 2002. A two-stage heuristic method for balancing mixed-model assembly lines with parallel workstations. *International Journal of Production Research*, 40, p. 1405-1420.

Villasenor, J. et Mangione-Smith, W.H., 1997. Configurable computing, Scientific American.

Wen-Chyuan, C., 1998. The application of a tabu search metaheuristic to the assembly line balancing problem. *Annals of Operations Research*, 77, p. 209-227.

White, W.W., 1961. Comments on a Paper by Bowman. *Operations Research*, 9, p. 274-276.

Wilhelm, W.E., 1999. A column-generation approach for the assembly system design problem with tool changes. *International Journal of Flexible Manufacturing Systems*, 11, p. 177-205.

Williams, H.P., 1993. Model Building in Mathematical Programming. John Wiley & Sons, New York.

Yalaoui, F. et Chu, C., 2002. Parallel machine scheduling to minimize total tardiness. International Journal of Production Economics, 76, p. 265-279.

Zhang, G.W., Zhang, S.C. et Xu, Y.S., 2002. Research on flexible transfer line schematic design using hierarchical process planning. *Journal of Materials Processing Technology*, 129, p. 629–633.

École Nationale Supérieure des Mines
de Saint-Étienne

N° d'ordre : 2010 EMSE 0588

Prénom NOM : Mohamed ESSAFI

DISSERTATION TITLE : Design and optimisation of resources allocation in reconfigurable machining lines

Speciality : : Industrial Engineering

Keywords : Reconfigurable transfer lines, line balancing, Combinatorial optimization, Mixed Integer Programming, Heuristics, Metaheuristics.

Abstract :

This work concerns the design and the optimization of reconfigurable transfer lines. The principle objective is to design a machining line with less cost while respecting the technological and economic constraints of the problem. The corresponding optimization problem is a transfer lines balancing problem subject to specific constraints. It consists to affect operations to workstations minimizing the installations cost. In addition to the habitual constraints of the transfer balancing problem, i.e. precedence, inclusion and exclusion constraints, we consider accessibility constraints. In addition, the principal specificity of reconfigurable lines compared to the dedicated transfer lines, comes from the sequential execution of operations. This often makes it necessary to set up stations with several machining centers working in parallel to achieve desired production volumes. Finally, the utilization of mono-spindle head machining center induces the inclusion of setup times between operations. This setup time is due to the time of displacement and change of tools which it depends of the operational sequence. We proposed firstly a mathematical formalization of the problem using a mixed integer program. We developed also several methods to calculate lower bounds and a pretreatment procedure. However, the additional constraints make the resolution of the considered balancing problem very difficult and the proposed approach generally does not solve instances of industrial size. To meet this need, we have developed several approximate resolution methods of the problem taking inspiration from effective Metaheuristics on combinatorial optimization problems.

École Nationale Supérieure des Mines
de Saint-Étienne

N° d'ordre : 2010 EMSE 0588
Prénom NOM : Mohamed ESSAFI

TITRE DE LA THÈSE : Conception et optimisation d'allocation de ressources dans les lignes d'usinage reconfigurables

Spécialité: : Génie Industriel

Mots clefs : : Lignes de transfert reconfigurables, Equilibrage des lignes, Optimisation combinatoire, Programmation linéaires en nombres entiers mixtes, Heuristiques, Métaheuristiques.

Résumé :

Les travaux de cette thèse concernent la conception et l'optimisation de lignes de transfert reconfigurables. L'objectif principal est de concevoir une ligne d'usinage à moindre coût tout en respectant les contraintes techniques, technologiques et économiques du problème. Le problème d'optimisation correspondant est un problème d'équilibrage de lignes d'usinage sujet à des contraintes spécifiques. Il consiste à affecter les opérations aux stations de travail en minimisant les coûts d'installation. En plus des contraintes habituelles de ce type de problème, à savoir, les contraintes de précédence, d'inclusion et d'exclusion, nous avons dû considérer des contraintes d'accessibilité. De plus, la spécificité principale des lignes reconfigurables par rapport aux lignes de transfert dédiées, vient de la réalisation en série des opérations. Celle-ci rend souvent nécessaire la mise en place de stations équipées de plusieurs centres d'usinage travaillant en parallèle pour obtenir les volumes de production souhaités. Enfin, l'utilisation d'une tête d'usinage mono-broche induit la prise en compte de temps inter-opératoire de déplacements et de changement d'outils qui dépendent de la séquence d'opérations. Dans un premier temps, nous avons proposé une modélisation mathématique du problème à l'aide d'un programme linéaire en nombres mixtes. Nous avons aussi développé des méthodes de calcul de bornes inférieures ainsi qu'une procédure de prétraitement. Cependant, les contraintes additionnelles rendent la résolution du problème d'équilibrage plus difficile que dans le cas des lignes dédiées, et l'approche proposée ne permet généralement pas de résoudre des instances de taille industrielle. Pour répondre à ce besoin, nous avons donc développé plusieurs méthodes de résolution approchées du problème en nous inspirant de métaheuristiques efficaces sur des problèmes d'optimisation combinatoire.