Conception d'un système d'aide à l'ordonnancement tenantcompte des impératifs économiques

Saad Ihsen

N° d'ordre : 50

ÉCOLE CENTRALE DE LILLE

UNIVERSITÉ DE TUNIS EL MANAR
ÉCOLE NATIONALE D'INGÉNIEURS DE TUNIS

THÈSE

présentée par

Ihsen SAAD
Ingénieur ENIG en Génie Électrique-Automatique

pour l'obtention du grade de

DOCTEUR

en Automatique et Informatique Industrielle

Doctorat délivré conjointement par l'École Centrale de Lille
et l'École Nationale d'Ingénieurs de Tunis

Conception d'un système d'aide à l'ordonnancement tenant compte des impératifs économiques

soutenue le 12 juillet 2007 devant le Jury d'Examen :

MM.	Pr.	Noureddine	ELLOUZE	Président
	Pr.	Naceur	BENHADJ BRAIEK	Rapporteur
	Pr.	Abdallah	EL MOUDNI	Rapporteur
	Pr.	Mohamed	BENREJEB	Examinateur
	Pr.	Pierre	BORNE	Examinateur
	Pr.	Slim	HAMMADI	Examinateur

Thèse préparée au Laboratoire d'Automatique, Génie Informatique et Signal de l'École Centrale de Lille
et à l'Unité de Recherche LARA-Automatique de l'École Nationale d'Ingénieurs de Tunis

Aux êtres les plus chers de l'univers :

A mes parents ;

Qui n'ont jamais cessé de m'aimer, de veiller à mon bonheur et de m'encourager pour accomplir mon devoir, qu'ils trouvent aujourd'hui dans ce travail le témoignage de mon profond attachement.

A mes sœurs, ma fiancée et tous ceux qui me sont chers avec toute mon affection.

A la mémoire de mon oncle.

Avant-propos

Le travail faisant l'objet du présent mémoire a été effectué au sein du Laboratoire de Recherche en Automatique (UR-LARA) de l'École Nationale d'Ingénieurs de Tunis (ENIT) et du Laboratoire d'Automatique, Génie Informatique et Signal (LAGIS) de l'École Centrale de Lille (EC-Lille).

Nous tenons à exprimer notre vive gratitude à Monsieur Nourredine ELLOUZE, Professeur et Directeur du Laboratoire de Systèmes et Traitement du Signal (LSTS) de l'ENIT, pour nous avoir fait le grand honneur d'accepter de présider le Jury d'Examen. Qu'il trouve ici l'expression de notre profond respect.

C'est un agréable devoir pour nous d'exprimer notre très vive reconnaissance à Monsieur le Professeur Mohamed BENREJEB, Directeur de l'Unité de Recherche UR-LARA à l'ENIT, à Monsieur Pierre BORNE, Professeur à l'EC-Lille, et à Monsieur Slim HAMMADI, Professeur à l'EC-Lille pour nous avoir guidé durant toute l'élaboration de ce mémoire avec le sérieux et la compétence qui les caractérisent. Qu'ils trouvent ici le témoignage de notre très profonde gratitude.

Nous tenons à remercier vivement Monsieur Naceur BEN HADJ BRAIEK, Professeur et Directeur de l'Unité de Recherche UR-LECAP à l'École Polytechnique de Tunis (EPT), d'avoir accepté de rapporter notre travail. Qu'il soit grandement remercié.

Notre profonde gratitude va à Monsieur Abdellah EL MOUDNI, Professeur à l'Université de Technologie Belfort-Monbéliard, pour avoir bien voulu rapporter sur nos travaux de thèse. Nous lui adressons nos sincères remerciements.

Nous tenons, enfin, à remercier tous les chercheurs du Laboratoire de Recherche en Automatique de l'ENIT et du Laboratoire d'Automatique, Génie Informatique et Signal de EC-Lille pour leur amicale présence et la sympathie qu'ils nous ont constamment témoignées. Nous leur exprimons ici toute notre gratitude.

Table des matières

Table des figures

Liste des tableaux

Glossaire

N	: nombre de produits
M	: nombre de machines
$N \times M$	: benchmark, N produits sur M machines
N_c	: ensemble de critères
n_c	: nombre de critères
j	: indice d'un produit
k	: indice d'une machine
i	: indice d'une opération
J_j	: produits finis
M_k	: k^{ieme} machine
S	: espace des solutions réalisables
n_j	: nombre d'opérations élémentaires du produit J_j
G_j	: gamme du produits J_j décrivant le séquencement des opérations élémentaires
$O_{i,j}$	: opération i du produit j, i étant son ordre dans la gamme, $1 \leq i \leq n_j$
r_j	: date de début au plus tôt du produit J_j
$r_{i,j}$	: date de début au plus tôt de l'opération $O_{i,j}$
d_j	: date de fin au plus tard à laquelle on désire avoir terminé l'exécution du produit J_j
tf_j	: date de fin d'exécution du produit J_j
$tf_{i,j,k}$	: date de fin d'exécution de l'opération $O_{i,j}$ sur la machine M_k
$r_{i,j,k}$	: date de début d'exécution de l'opération $O_{i,j}$ sur la machine M_k
$p_{i,j,k}$	: durée d'exécution de l'opération $O_{i,j}$ sur la machine M_k
C_{max}	: Makespan

$\alpha_{i,j}$	: durée minimale de $O_{i,j}$, $\alpha_{i,j} = \min_{1 \leq k \leq M}(d_{i,j,k})$,
w_k	: charge de la machine M_k
$\omega_{j,k}$	: coefficient d'utilisation de la machine M_k pour la production du produit J_j, $\omega_{j,k} \in \{0, 1\}$
w_{kj}	: charge de la machine M_k pour le produit J_j
Mp_j	: matière première du produit J_j
Cm_k	: coût unitaire sur la machine M_k
h_j	: coût de pénalisation du produit J_j pour l'avance (stockage)
b_j	: coût de pénalisation du produit J_j pour le retard
A_j	: coût de la pénalité sur le produit J_j
Cf_j	: coût de fabrication du produit J_j
Cp_j	: coût de la production totale d'un produit J_j
C_p	: coût de la production totale
μ	: mesure floue sur N_c
$\mu(A)$	: importance ou pouvoir de groupe de critères A
a_i	: valeur du critère i
a_1, ..., a_{n_c}	: scores d'une solution suivant tout critère
a	: vecteur des critères
$C(x)$	: vecteur des valeurs de critères, associé à la solution x
$C_i(x)$	: valeur du critère i associé à la solution x
C_i	: i critère à optimiser
C_i^b	: borne inférieure du critère i
C_i^h	: borne supérieure du critère i
B^i	: sous-ensemble flou caractérisant les bonnes solutions par rapport au critère i
M^i	: sous-ensemble flou caractérisant les mauvaises solutions par rapport au critère i
x	: solution réalisable
$\mu_i^B(C_i(x))$	: mesure floue de $C_i(x)$ dans le sous-ensemble B^i
$C_B(x)$	: vecteur de la qualité de la solution x
$C_g(x)$	: évaluation globale de la solution
I_i	: indice de Shapley (indice d'importance) pour le i^{ieme} critère
$I_{i,j}$	: indice d'interaction entre le critère i et le critère j

I	: matrice d'interaction
$C_\mu(a)$	: intégrale de Choquet pour le vecteur a
$C_{OWA}(a)$	: valeur d'agrégation OWA (Ordered Weighted Averaging Operators) pour le vecteur a
w_i	: poids d'agrégation du critère i

Introduction Générale

Les travaux présentés dans cette thèse concernent la résolution du problème d'ordonnancement dans un job-shop flexible. Une première étude nous a conduit à traiter le problème en supposant les machines toujours disponibles et en s'intéressant à l'optimisation du Makespan C_{max}, de la charge critique, de la charge totale, de la somme des retards et du coût de la fabrication, aspect peu étudié pour ce type de problème. Dans un deuxième temps, nous avons tenu compte du problème de décision et supposé que les opérations de production sont strictement non-préemptives ; ce qui signifie que l'exécution d'une opération ne peut être interrompue par la réalisation d'une tâche.

Malgré son apparente simplicité, la plupart des problèmes à une machine sont NP-difficiles. L'étude de ce type de problèmes est utile à la compréhension et la modélisation des problèmes à plusieurs machines. C'est pour cela que plusieurs chercheurs y ont investi beaucoup d'effort dans le sens du traitement des cas les plus rencontrés dans le milieu industriel. Le travail de recherche que nous présentons traite un problème d'ordonnancement multi-objectifs à plusieurs machines. Ce problème consiste à ordonnancer la fabrication de produits sur plusieurs machines en tenant compte du coût de la production et de la somme des retards.

La formulation mathématique a été précédée d'un état de l'art des différents travaux pouvant nous aider. Cette formulation a été présentée en détaillant les différents critères à optimiser, avant d'aborder la résolution du problème traité.

Compte tenu de l'outil de résolution adopté, nous proposons deux nouvelles approches d'évaluations multi-critères. Basées sur le principe de la logique floue et

des bornes inférieures, celles-ci approches visent l'homogénéisation des critères en vue d'harmoniser leurs valeurs numériques afin que l'un ne domine pas l'autre. L'adoption de ces approches permet d'offrir au décideur un ensemble de solutions réalisables. Après, les approches évolutionnistes élaborées, plusieurs résultats de simulation sont présentés afin de montrer les performances des approches proposées.

Dans le premier chapitre de ce mémoire, nous situons notre travail dans le cadre de l'ordonnancement des systèmes de production. Pour cela, nous présentons la problématique de l'ordonnancement et rappelons, en premier lieu, les différents éléments qui composent un problème d'ordonnancement ainsi que les notations utilisées permettant de le caractériser. Nous présentons en second lieu une typologie des problèmes d'ordonnancement qui permet de distinguer les différents types d'ateliers. Cette classification nous amène à étudier la complexité des problèmes rencontrés. Notre intérêt se focalise ensuite sur les méthodes de résolution développées dans la littérature. En particulier, nous donnons une description détaillée de ces méthodes, essentiellement les algorithmes génétiques, utilisés et leurs applications aux problèmes d'ordonnancement.

Avant d'aborder l'étude du problème posé, nous avons formulé les différents critères étudiés ainsi que leurs bornes inférieures dans le chapitre 2. L'approche proposée, à base de l'intégrale floue de Choquet, consiste à résoudre tout d'abord le problème d'affectation puis le problème de décision d'une manière statique.

En effet, pour le problème de minimisation, nous avons cherché à optimiser le Makespan, la charge de la machine critique (machine la plus chargée), la charge totale des machines, les pénalités de retard et de stockage ainsi que le coût de la production.

Des bornes inférieures pour les différents critères ont été définies afin de donner une estimation du critère étudié et de mesurer ainsi la qualité de l'affectation des opérations aux machines. Le problème de séquencement a été résolu en utilisant un algorithme génétique.

Notre méthode a été validée en effectuant des comparaisons avec d'autre types d'approches de la littérature, utilisées pour la résolution de ce problème.

Le chapitre 3 concerne l'étude du job-shop flexible par l'agrégation dynamique des

critères. Dans un premier temps, nous avons développé une méthode d'agrégation avec direction de recherche dynamique tenant compte du coût de production et utilisant les critères définis dans le chapitre 2. Une approche intégrée, basée sur l'ε-dominance, est également présentée, une borne inférieure ainsi qu'une adaptation de l'algorithme génétique utilisé pour résoudre le problème classique sont proposées pour le problème général.

A la fin de ce chapitre, ces méthodes ont été validées en effectuant des comparaisons avec d'autres types d'approches.

Chapitre 1

Problème d'ordonnancement : caractérisation et typologie

Sommaire

1.1 Introduction aux problèmes d'ordonnancement

Les problèmes d'ordonnancement sont présents dans tous les secteurs d'activité de l'économie depuis l'industrie manufacturière (Pinedo [Pinedo, 1955]) jusqu'à l'informatique (Blazewicz et al. [Blazewicz et al., 1996]). C'est pour cette raison qu'ils ont fait et continuent de faire l'objet de nombreux travaux de recherche. Citons par exemple ceux de Conwway et al. [Conway et al., 1967], Backer [Backer, 1974], Carlier et Chretienne [Carlier et Chretienne, 1988], Brucker [Brucker, 1998], Esquirol et Lopez [Lopez et Esquirol, 1999].

L'ordonnancement est une branche de la recherche opérationnelle et de la gestion de production qui vise à améliorer l'efficacité d'une entreprise en termes de coût de production et de délai de livraison.

Résoudre un problème d'ordonnancement consiste à ordonnancer i.e. à programmer ou à planifier, dans le temps l'exécution des tâches en leur attribuant les ressources matérielles ou humaines nécessaires de manière à satisfaire un ou plusieurs critères préalablement définis, tout en respectant les contraintes de réalisation (GOTHA [GOTHA, 1993] ; Lopez et Roubellat [Lopez et Roubellat, 2001]).

Ainsi le résultat d'un ordonnancement est un calendrier précis des tâches à réaliser. Il se décompose en trois grandeurs fondamentales :

- l'affectation qui consiste à attribuer les ressources nécessaires à une tâche ;
- le séquencement qui précise l'ordre de passage des tâches sur chaque ressource ;
- le datage qui donne pour chaque tâche une date de début et une date de fin.

Un ordonnancement se décompose en deux parties toujours présentes dans l'atelier, mais pouvant revêtir une importance variable :

- l'ordonnancement prédictif qui consiste à prévoir a priori un certain nombre de décisions en fonction de données prévisionnelles et d'un modèle de l'atelier ;
- l'ordonnancement réactif qui consiste à adapter les décisions prévues en fonction de l'état courant du système et des déviations entre la réalité et le modèle théorique.

La maîtrise de l'ordonnancement est d'un intérêt capital pour les entreprises, sans

cesse confrontées a des impératifs de productivité, de flexibilité et de réactivité. L'étude des problèmes d'ordonnancement est également d'un intérêt théorique toujours renouvelé pour les chercheurs, car à ce jour il n'existe pas encore de méthode de résolution à la fois générale et de faible complexité algorithmique, à cause de la nature fortement combinatoire de ces problèmes. Le contexte récent de la production flexible, où une machine donnée peut usiner un nombre important de pièces différentes, rend indispensable la résolution rigoureuse des problèmes d'ordonnancement et d'affectation qui en découlent.

Dans ce travail, nous nous intéressons à l'ordonnancement prédictif d'un atelier de type job-shop flexible. Nous introduisons, dans un premier temps dans ce chapitre, quelques généralités sur les problèmes d'ordonnancement dans les ateliers de production à savoir les types d'ateliers, les critères d'optimisation ainsi qu'une brève description de la notion de complexité des problèmes d'optimisation combinatoire. Dans un deuxième temps est réalisée une description des principales méthodes de résolution des problèmes d'ordonnancement proposées dans la littérature, plus précisément la procédure de séparation et d'évaluation, les approches par recherche locale et les algorithmes génétiques. L'accent est ensuite mis sur l'application des algorithmes génétiques aux problèmes d'ordonnancement.

1.2 Ordonnancement dans un atelier de production

1.2.1 Éléments d'un problème d'ordonnancement

Regardons tout d'abord plus en détail comment s'énonce classiquement un problème d'ordonnancement. Il faut fixer à la fois les caractéristiques des tâches, les liens entre elles (les contraintes), les caractéristiques des ressources et le critère de performance choisi.

Définition 1.1 *Ordonnancer un atelier, consiste à programmer l'exécution des*

opérations élémentaires (tâches ou jobs) en leur allouant les ressources requises et en fixant leurs dates de début de fabrication [Carlier et Chretienne, 1988]. Il s'agit donc essentiellement des deux décisions [Chu et Proth, 1996] suivantes :

- *affecter chaque opération à une ressource : choisir une ressource parmi un ensemble de ressources,*
- *déterminer le séquencement des opérations attribuées à une même ressource en présentant leur ordre de passage ainsi que leurs dates de début d'exécution.*

Cette allocation d'opérations-ressources est réalisée en tenant compte des contraintes temporelles définies par la planification, des charges et de la disponibilité des ressources imposées par la programmation prévisionnelle de la production, des contraintes de succession des opérations imposées par la gamme de fabrication, et d'une fonction coût de l'ordonnancement à minimiser (retard, temps global, nombre de retards,...) [Liouane, 1998].

1.2.1.1 Les opérations

Le plan directeur de production communiqué aux agents de maîtrise de l'atelier précise un ensemble de produits finis $\{ J_j, j = 1, .., n \}$ à réaliser. La fabrication de chaque produit nécessite l'exécution d'un nombre n_j d'opérations élémentaires dont le séquencement est décrit par une gamme G_j. Une opération est notée $O_{i,j} : j$ étant le produit dont elle fait partie et i son ordre dans la gamme avec $1 \leq i \leq n_j$. Elle est caractérisée par [Gargouri, 2003] :

- sa date de disponibilité (réalisabilité) $r_{i,j}$, appelée aussi date de début au plus tôt ;
- sa durée opératoire $p_{i,j}$ qui dépend de la machine ou de la ressource opératrice ;
- sa date échue ou encore date de fin au plus tard $d_{i,j}$.

Les opérations à réaliser sont souvent liées entre elles par des relations d'antériorité. Si ce n'est pas le cas, les opérations sont dites *indépendantes*.

Dans certains cas, une opération peut être exécutée par morceaux. Elle est dite, alors, *morcelable* ou encore *préemptive*. Dans le cas contraire, une opération ne peut être interrompue avant son achèvement. D'autres opérations peuvent être interrompus et doivent être reprises dès le début ; on parle alors d'opérations *préemptives répétées*.

1.2.1.2 Les ressources

Une ressource est un moyen technique ou humain utilisé pour réaliser une opération. Il existe deux types de ressources qui sont [Gargouri et al., 1999] :

– les *ressources consommables* telles que les matières premières ;
– les *ressources renouvelables*, c'est à dire pouvant être réutilisées après la fin de l'exécution des opérations comme les machines, le personnel,... Ces ressources peuvent aussi être classées en :
  ▪ ressources de type *disjonctif* qui traitent seulement une opération à la fois ;
  ▪ ressources du type *cumulatif* qui peuvent exécuter plusieurs opérations simultanément.

1.2.1.3 Les contraintes

Ce sont des conditions à respecter dans la construction de l'ordonnancement pour qu'il soit réalisable.

Suivant leur lien avec le système de production, les contraintes peuvent être classées selon deux types [Kacem, 2003] : endogènes et exogènes.

✓ **Les contraintes endogènes :** ce sont les conditions directement liées au système de production et à ses performances. Nous pouvons citer :
  – les capacités des machines et des moyens de transports ;
  – les dates de disponibilité des machines et des moyens de transport ;
  – les séquences des actions à effectuer ou les gammes des produits.

✓ **Les contraintes exogènes :** ce sont les contraintes imposées extérieurement. Elles sont indépendantes du système de production ; on distingue celles relatives :
  – aux dates dues pour chaque produit imposées généralement par les commandes ;
  – aux priorités de quelques commandes et de quelques clients ;
  – et aux retards accordés pour certains produits.

Une autre classification est possible selon la disponibilité des ressources et suivant l'évolution temporelle [Kacem, 2003].

✓ **Les contraintes de ressources** : plusieurs types de contraintes peuvent être

induites par la nature des ressources. A titre d'exemple, la capacité limitée d'une ressource implique qu'on ne peut pas dépasser un certain nombre de tâches exécutées sur cette ressource. L'hypothèse disjonctive induit une contrainte de réalisation de tâches sur des intervalles temporels disjoints pour une même ressource. Néanmoins, la nature cumulative d'une ressource implique généralement la limitation du nombre des tâches à réaliser en parallèle.

✓ **Les contraintes temporelles** : elles représentent des restrictions sur les valeurs que peuvent prendre certaines variables temporelles d'ordonnancement. A titre d'exemple, nous présentons une liste non exhaustive de contraintes souvent rencontrées :

- les *contraintes des dates dues* : lorsque certaines tâches doivent être achevées avant une date butoir préalablement fixée ;
- les *contraintes temporelles de précédence* : lorsque la tâche (i) doit précéder la tâche $(i+1)$, exprimant qu'une contrainte temporelle que l'on peut traduire par l'inégalité suivante : $tf_i \leq t_{i+1}$, avec tf_i la date de fin d'exécution de la tâche (i) et t_{i+1} la date de début d'exécution de la tâche $(i+1)$;
- les *contraintes des dates au plus tôt* : qui sont liées à l'indisponibilité de certains facteurs nécessaires pour commencer l'exécution des tâches. Ce type de contrainte peut être exprimée pour une tâche (i) par : $t_i \geq r_i$ où t_i est la date de début d'exécution de la tâche (i) et r_i la date de début au plus tôt de la même tâche.

1.2.1.4 Le(s) critère(s) d'optimisation

Un problème d'ordonnancement n'est pas nécessairement exprimé comme un problème d'optimisation. Néanmoins, la notion de critère d'optimisation est toujours présente, au moins implicitement. Un chef d'atelier veut pouvoir terminer le plan de production journalier dans les délais, imposés. Un chef de projet désire assurer une charge de travail la plus constante possible à son équipe durant le déroulement du projet, dont la durée est fixée. Un programmeur utilisant de lourdes ressources de

calcul désire que son application soit correctement exécutée, et que le résultat lui parvienne au bout d'un temps raisonnable. Ainsi Les critères possibles, dépendant de l'application considérée, sont très nombreux. Il peut même y en avoir plusieurs à la fois.

Parmi les ordonnancements possibles, nous devons choisir la solution la plus satisfaisante. Cette notion de satisfaction dépend du critère préalablement défini. Les critères sont donc les barèmes qui vont servir à l'évaluation et à la comparaison de l'ensemble des ordonnancements possibles. Ces critères peuvent être classés en deux types [Kacem, 2003] : réguliers et irréguliers.

- **Les critères réguliers** : les critères sont dits réguliers car ils constituent des fonctions décroissantes des dates d'achèvement des opérations. Nous citons à titre d'exemple :
  - la minimisation des dates d'achèvement des actions ;
  - la minimisation du maximum des dates d'achèvement des actions ;
  - la minimisation de la moyenne des dates d'achèvement des actions ;
  - la minimisation des retards sur les dates d'achèvement des actions ;
  - la minimisation du maximum des retards sur les dates d'achèvement des actions ;
  - la minimisation de la moyenne des retards sur les dates d'achèvement des actions ;
  - la minimisation du temps du cycle (cas d'ordonnancement cyclique).

- **Les critères irréguliers** : ce sont les critères non réguliers, c'est à dire qui ne sont pas des fonctions monotones des dates de fin d'exécution des opérations. Soit à titre d'exemple :
  - la minimisation des encours ;
  - la minimisation du coût du stockage des matières premières ;
  - l'équilibrage des charges des machines.

La satisfaction de tous les critères à la fois est souvent délicate car on se trouve, généralement, devant des situations contradictoires [Roy et Bouyssou, 1993].

1.2.1.5 Les classes d'ordonnancement

Il existe différentes classes d'ordonnancement que nous pouvons définir comme suit.

- Dans un ordonnancement semi-actif, il est impossible d'avancer une opération sans modifier la séquence des opérations sur la ressource : chaque opération est calée, soit sur l'opération qui la précède dans sa gamme, soit sur l'opération qui la précède sur la machine utilisée.
- Dans un ordonnancement actif, il est impossible d'avancer une opération sans reporter le début d'une autre opération.
- Un ordonnancement est sans délai ou sans retard si et seulement si aucune opération n'est mise en attente, alors qu'une machine est disponible pour l'exécuter.

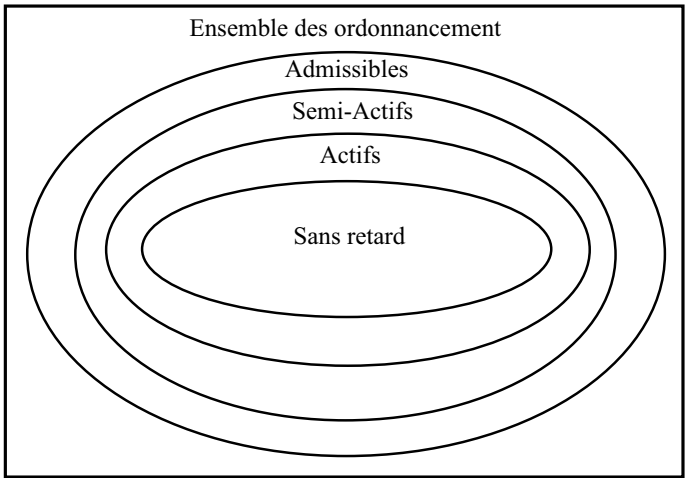

FIG. 1.1 – Classement des ordonnancements

La figure 1.1 représente le diagramme d'inclusion des classes d'ordonnancements. Elle fait apparaître que les ordonnancements sans retard sont inclus dans le sous-ensemble des ordonnancements actifs qui sont eux-mêmes inclus dans le sous-ensemble des ordonnancements semi-actifs.

La propriété 1 suivante (Backer [Backer, 1974]) lie les critères réguliers à la classe des ordonnancements actifs.

Propriété 1 *L'ensemble des ordonnancements semi-actifs est dominant dans les*

problèmes d'optimisation d'un critère régulier et le sous-ensemble des ordonnancements actifs est le plus petit ensemble dominant.

En conséquence, dans le cas d'un critère régulier, la recherche d'une solution optimale peut être limitée à l'ensemble des ordonnancements actifs ; ce qui restreint ainsi la taille de l'espace de recherche.

1.2.2 Problèmes d'ordonnancement

Une classification des problèmes d'ordonnancement peut s'opérer selon le nombre de machines et leur ordre d'utilisation pour fabriquer un produit (gamme de fabrication : enchaînement logique des opérations) qui dépend de la nature de l'atelier. Un atelier est caractérisé par le nombre de machines qu'il contient et par son type. Pour les différents types d'ateliers possibles, différents problèmes peuvent être posés [Carlier et Chretienne, 1988].

- *Problème à une machine* : chaque produit est constitué d'une seule opération. L'intérêt de ce type de problème réside dans le fait qu'il permet de développer des méthodes utilisables pour la résolution de problèmes plus complexes.

- *Problème à machines parallèles* : elles remplissent, a priori, toutes les mêmes fonctions. Selon leur vitesse d'exécution, sont utilisées :
  - des machines identiques : la vitesse d'exécution est la même pour toutes les machines et pour tous les produits ;
  - des machines uniformes : chaque machine a une vitesse d'exécution propre et constante. La vitesse d'exécution est la même pour tous les produits d'une même machine ;
  - des machines indépendantes : la vitesse d'exécution est différente pour chaque machine et pour chaque produit.

- *Problème de flow-shop* : il se rencontre dans les ateliers disposant de lignes de production dédiées à la production de masse de peu de variétés de produits. Dans

ce type d'atelier, les produits à réaliser sont composés de plusieurs opérations et visitent toutes les machines dans le même ordre (Figure 1.2).

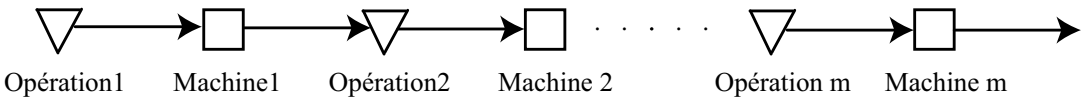

FIG. 1.2 – Atelier de type flow-shop

La littérature présente une extension possible de ce problème : le flow-shop hybride où les machines sont disponibles en plusieurs exemplaires.

– *Problème de job-shop* : il constitue une généralisation directe de celui du flow-shop. En effet, les opérations élémentaires d'un même produit suivent une séquence d'exécution totalement ordonnée et variable selon le produit (Figure 1.3). Le modèle de job-shop est exploité dans le cadre de cette étude.

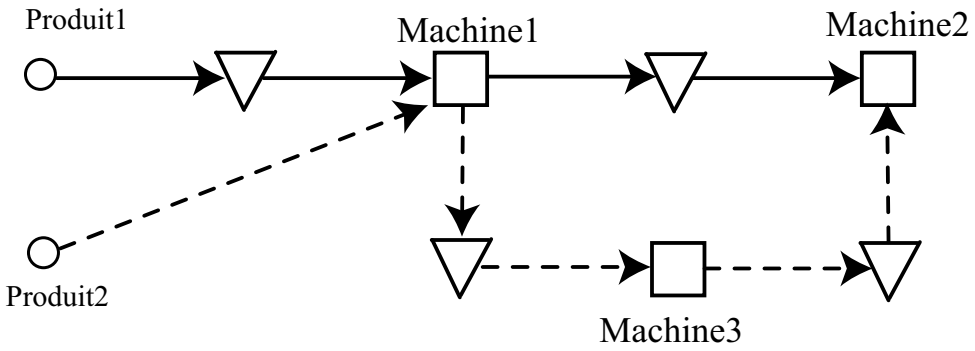

FIG. 1.3 – Atelier de type job-shop

Une extension de ce modèle est le job-shop flexible. Sa particularité est que chaque opération peut être exécutée sur plusieurs machines. Il s'ensuit qu'il offre plus de flexibilité par rapport au job-shop classique grâce à la polyvalence des machines. Toutefois, cela induit une complexité supplémentaire due à la nécessité de la détermination des affectations adéquates avant d'établir l'ordre de passage des différentes tâches sur les machines. Ce type de problème est traité dans ce mémoire.

– *Problème de l'open-shop* : c'est un modèle d'atelier moins contraint que le flow-shop et le job-shop, car l'ordre des opérations n'est pas fixé a priori. Le problème d'ordonnancement consiste d'une part à déterminer le cheminement de chaque produit et d'autre part à ordonnancer les produits en tenant compte des gammes trouvées. Notons que ces deux problèmes peuvent être résolus simultanément. Comparé aux autres modèles d'ateliers multi-machines, l'open-shop n'est pas couramment utilisé dans les entreprises.

1.2.3 Modélisation et représentation des problèmes d'ordonnancement

La modélisation est généralement une étape très importante de la résolution d'un problème. C'est une écriture simplifiée de toutes les données tout en utilisant un formalisme bien adapté pour représenter un problème choisi. Dans la littérature, on trouve deux méthodes différentes pour modéliser les problèmes d'ordonnancement : les méthodes mathématiques et les méthodes graphiques.

1.2.3.1 La programmation mathématique

Ces méthodes consistent à représenter les données du problème (contraintes et fonction économique) sous forme d'équations et inéquations mathématiques. Ces méthodes, très couramment utilisées, ont l'avantage d'être simples d'une part et directement exploitables par les algorithmes de résolution d'autre part [Lopez et Esquirol, 1999].

Exemple 1.2.1 *(figure 1.4) Pour* $i \in \{1, 2, 3, 4, 5, 6, 7\}$*, il est question de calculer* t_i *la date de début d'exécution de la tâche* i *en minimisant le* C_{max} *et en respectant les contraintes suivantes,* d_i *étant la durée opératoire de la tâche (i) :*

Contraintes des données	***Contraintes de précédence***	***Contraintes des ressources***
$d_1 = 3$	$t_1 + d_1 \leq t_3$	$(t_1 + d_1 \leq t_2) OU (t_2 + d_2 \leq t_1)$
$d_2 = 2$	$t_1 + d_1 \leq t_4$	
$d_3 = 4$		$(t_3 + d_3 \leq t_5) OU (t_5 + d_5 \leq t_3)$
$d_4 = 1$	$t_3 + d_3 \leq t_6$	$(t_3 + d_3 \leq t_7) OU (t_7 + d_7 \leq t_3)$
$d_5 = 8$	$t_4 + d_4 \leq t_7$	$(t_7 + d_7 \leq t_5) OU (t_5 + d_5 \leq t_7)$
$d_6 = 3$	$t_2 + d_2 \leq t_5$	
$d_7 = 7$		$(t_4 + d_4 \leq t_6) OU (t_6 + d_6 \leq t_4)$

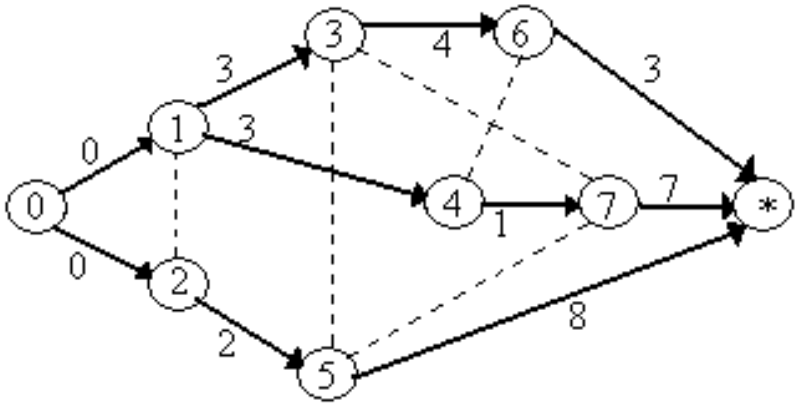

FIG. 1.4 – Modélisation par graphe

1.2.3.2 Les graphes

Pour modéliser un problème d'ordonnancement, nous pouvons avoir recours aux graphes. Dans ce genre de modélisation, les tâches sont représentées par des nœuds, les contraintes de précédence par des arcs conjonctifs tout en indiquant les durées des tâches et finalement les contraintes de ressources par des arcs disjonctifs [Roy, 1970][GOTHA, 1993].

Un problème d'ordonnancement peut être représenté par un graphe potentiel-tâches noté $G(X, U)$ où X est l'ensemble des sommets et U l'ensemble des arcs.

Si on appelle T l'ensemble des n tâches du problème considéré, X est alors la réunion de T et de l'ensemble $\{0, n+1\}$ où (0) représente la tâche du début fictif et (n+1) la tâche de fin fictive.

L'ensemble des arcs U est $\{(0,i), (i,j)$ et $(i,n+1)$ tels que i et $j \in T\}$. Cet ensemble représente les différentes contraintes reliant les tâches (figure 1.5).

Les valeurs portées par les arcs dépendent des sommets :

⋆ les arcs de types $(0,i)$: l'arc porte la date de disponibilité de la tâche i ;

⋆ les arcs de types (i,j) : l'arc porte la valeur de la durée opératoire de la tâche i. Un deuxième arc portant une valeur négative peut éventuellement relier j et i pour visualiser le fait que la tâche j doit commencer immédiatement après la tâche i ;

⋆ les arcs de type (i,n) : la valeur portée par l'arc est la durée opératoire de i.

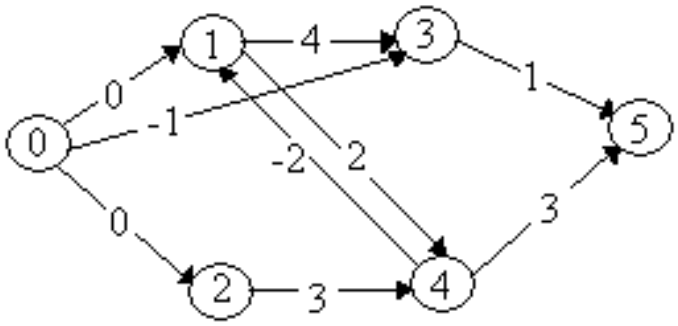

FIG. 1.5 – Graphe Potentiel-Tâches

Cette méthode graphique a été développée grâce à la théorie des Réseaux de Pétri (RdP) qui ont surtout servi à modéliser les systèmes dynamiques à événements discrets [Carlier et al., 1984]. Ces RdP ont l'avantage de pouvoir visualiser par des transitions les changements d'état d'un système.

Exemple 1.2.2 *La figure 1.4 constitue la modélisation d'un problème d'ordonnancement de 7 tâches à exécuter sur 3 ressources. Les contraintes de précédence sont les suivantes :* $\{$*1 avant 3 ; 3 avant 6 ; 1 avant 4 ; 4 avant 7 ; 2 avant 5*$\}$*. Les durées des tâches sont indiquées sur les arcs conjonctifs (par exemple la durée de la tâche 1 est de 3 unités de temps). Les tâches 1 et 2 doivent être réalisées sur la première*

ressource. Les tâches 3, 5 et 7 doivent être réalisées sur la deuxième ressource. La troisième ressource exécutera les tâches 4 et 6. Ainsi, les contraintes sur les ressources sont représentées par les arêtes pointillées. Le fait de choisir un sens pour ces arêtes donne un ordonnancement possible.

1.2.4 Complexité des problèmes

D'une manière générale, les problèmes d'ordonnancement d'atelier sont des problèmes combinatoires extrêmement difficiles et il n'existe pas de méthodes universelles permettant de résoudre efficacement tous les cas.

- **Complexité méthodologique** : Elle exprime une fonction du nombre d'opérations élémentaires de calcul effectuées par la méthode ou par l'algorithme de résolution en fonction du nombre des données du problème traité.

- **Complexité problématique** : Cette notion est liée à la difficulté du problème à résoudre et au nombre des opérations élémentaires qu'un algorithme déterministe doit effectuer pour trouver l'optimum en fonction de la taille du problème. Selon son degré de complexité, un problème peut appartenir à l'une des quatre classes suivantes [Sakarovitch, 1984].

  - *Les problèmes les plus difficiles* : ce sont les problèmes pour lesquels il n'existe aucune méthode de résolution. Ils sont également dits indécidables.

  - *Les problèmes de la classe P* : un problème de décision est dit polynomial s'il existe un algorithme de complexité polynômiale pour sa résolution. Mais, ne pas connaître un tel algorithme ne signifie pas forcément qu'il n'existe pas.

  - *Les problèmes de la classe NP* : ils sont dits NP-difficiles. Ces problèmes ne peuvent a priori être résolus en un temps polynomial que par des méthodes approchées (appelés également heuristiques). Au cours de leur exécution, ces algorithmes font des choix dont l'optimalité n'est pas démontrable.

– *Les problèmes NP-complets* : un problème de décision A est dit NP-complet s'il appartient à la classe NP et si pour tout A' de NP, on a :

  ⋆ il existe une application polyômiale qui transforme toute instance I' de A' en une instance I de A.
  ⋆ A' admet une réponse "oui" pour l'instance I', si et seulement si A admet une réponse "oui" pour l'instance I.

  Autrement dit, s'il existe un algorithme polynomial pour résoudre A, alors, pour tout le reste des problèmes de la classe, il existe des algorithmes polynômiaux pour leur résolution.
  On peut voir de façon intuitive les problèmes NP-Complets comme des problèmes pour lesquels la recherche d'une solution consiste à parcourir un arbre. Cet arbre de recherche contient l'ensemble des solutions possibles. Une branche de cet arbre représente une éventuelle solution. La hauteur d'un tel arbre est polynômiale alors que, son nombre de branches est par contre exponentiel.

1.2.5 Méthodes de résolution

De nombreuses méthodes de résolution ont été développées en Recherche Opérationnelle (RO) et en Intelligence Artificielle (IA). Ces méthodes peuvent être classées sommairement en deux grandes catégories : les méthodes exactes (complètes) qui garantissent la complétude de la résolution et les méthodes approchées (incomplètes) qui perdent la complétude pour gagner en efficacité.

1.2.5.1 Les méthodes exactes

On peut définir une méthode exacte comme une méthode qui fournit une solution optimale pour un problème d'optimisation. L'utilisation de ces méthodes s'avère particulièrement intéressante dans le cas des problèmes de petite taille [Sakarovitch, 1984].

Le principe essentiel d'une méthode exacte consiste généralement à énumérer, souvent de manière implicite, l'ensemble des solutions de l'espace de recherche. Pour améliorer l'énumération des solutions, une telle méthode dispose de techniques pour détecter le plus tôt possible les échecs (calculs de bornes) et d'heuristiques spécifiques pour orienter les différents choix. Parmi les méthodes exactes, on trouve la plupart des méthodes traditionnelles (développées depuis une trentaine d'années) telles les techniques de Séparation et Evaluation Progressive (SEP) ou les algorithmes avec retour arrière. Les méthodes exactes ont permis de trouver des solutions optimales pour des problèmes de taille raisonnable. Malgré les progrès réalisés (notamment en matière de programmation linéaire en nombres entiers), comme le temps de calcul nécessaire pour trouver une solution, risquent d'augmenter exponentiellement avec la taille du problème, les méthodes exactes rencontrent généralement des difficultés face aux applications de taille importante [Hao et al., 1999].

Procédure de Séparation et d'Evaluation (PSE)

Les procédures par séparation et évaluation sont également appelées "branch and bound" [Sakarovitch, 1984], elles consistent à décomposer l'ensemble des solutions en sous-ensembles de solutions partielles. Le calcul d'une borne inférieure pour chaque solution partielle permet d'évaluer sa qualité. Si cette borne inférieure est supérieure ou égale à la meilleure borne trouvée, on peut couper la branche correspondant à la solution partielle car il n'existe pas de meilleure solution dans ce sous-ensemble. La difficulté de cette méthode réside dans le calcul d'un minorant suffisamment bon pour éliminer le plus rapidement possible un maximum de branches et atteindre un optimum.

Pour les problèmes d'ordonnancement de grande taille, cette technique risque de générer une arborescence conséquente à explorer. Pour remédier à un tel problème, la borne utilisée doit être la plus fine possible [Carlier et Chretienne, 1988].

L'algorithme de Johnson

L'algorithme de Johnson est fondé sur le calcul itératif du minimum des durées

des tâches associées aux travaux non encore placés. Cet algorithme donne une solution optimale pour le problème à deux machines de type flow shop. Nous présentons ci-dessous l'algorithme de Johnson [Carlier et Chretienne, 1988].

Début

$U = \emptyset$; $V = \emptyset$

Pour (i=1 jusqu'à n) **faire** (n est le nombre total d'opérations)

Si ($a_i < b_i$) **alors**

U=U+{i}

Sinon

V=V+{i}

Fin Pour

LU : liste ordonnée par a_i croissants au sens large des éléments de U

LV : liste ordonnée par b_i croissants au sens large des éléments de V

(où a_i est le temps d'exécution de l'opération i sur la machine 1

et b_i est le temps d'exécution de l'opération i sur la machine 2)

L = (LV) (LU) (concaténation des listes LU et LV)

(L représente la séquence optimale).

Fin.

Programmation dynamique

Elle se base sur le principe de Bellman [Borne et al., 1990] : « Si C est un point qui appartient au chemin optimal entre A et B, alors la portion de ce même chemin allant de A à C est le chemin optimal entre A et C ». Elle consiste donc à construire d'abord les sous chemins optimaux et ensuite par récurrence le chemin optimal pour le problème entier. Cette méthode est destinée à résoudre des problèmes d'optimisation à vocation plus générale que la méthode de séparation et évaluation. Par contre, elle ne permet pas d'aborder des problèmes de taille aussi importante que le « branch and bound ».

Programmation linéaire en nombres entiers

C'est l'une des techniques classiques de la recherche opérationnelle. C'est un cas particulier de la procédure de séparation et d'évaluation dont l'évaluation est basée sur une relaxation réelle. Cette méthode repose sur les travaux du simplexe et les algorithmes des points intérieurs de Karmarkar [Kacem, 2003].

Elle consiste à minimiser une fonction coût en respectant des contraintes. Le critère et les contraintes sont des fonctions linéaires des variables du problème.

1.2.5.2 Les heuristiques

Les heuristiques sont des méthodes empiriques (qui donnent généralement de bons résultats sans être démontrables). Elles se basent sur des règles simplifiées pour optimiser un ou plusieurs critères. Le principe général de ces méthodes est d'intégrer des stratégies de décision pour construire une solution proche de l'optimum.

Nous exposons ci-dessous les plus utilisées d'entre elles [Chu et Proth, 1996].

- **FIFO (First In First Out) ou FCFS (First Come First Served) :** La première tâche qui vient est la première tâche ordonnancée.
- **SPT (Shortest Processing Time) :** La tâche ayant le temps opératoire le plus court est traitée en premier lieu.
- **LPT (Longest Processing Time) :** La tâche ayant le temps opératoire le plus important est ordonnancée en premier lieu.
- **EDD (Earliest Due Date) :** La tâche ayant le date due la plus petite est la plus prioritaire.
- **SRPT (Shortest Remaining Processing Time) :** Cette règle sert à lancer la tâche ayant la plus courte durée de travail restant à exécuter. Elle est très utilisée pour minimiser les encours et dans le cas des problèmes d'ordonnancement préemptifs.
- **ST (Slack Time) :** A chaque point de décision, l'opération ayant la plus petite marge temporelle est prioritaire. Faute de disponibilité des ressources de production, cette marge peut devenir négative.

1.2.5.3 Les méthaheuristiques

Elles représentent des concepts généraux de résolution. Il s'ensuit qu'il faut formuler le problème abordé de telle manière qu'il soit adapté à l'application de ces concepts. L'avantage de ces méthodes consiste en la réduction de temps du calcul. Elles permettent d'obtenir un bon compromis entre la qualité des solutions et le temps de calcul.

Nous trouvons, dans cette catégorie, les algorithmes génétiques, les réseaux de neurones, la méthode tabou, le recuit simulé, la logique floue, etc., qui forment une sorte de lien entre plusieurs disciplines différentes et l'ordonnancement. Nous présentons, dans la suite, les approches les plus connues avec leurs spécificités.

1. ***Méthode de recherche locale***

   La recherche locale est une technique très utilisée pour la résolution des problèmes d'optimisation combinatoire. Cette méthode est itérative et consiste à se déplacer dans l'espace de recherche d'une solution à une autre meilleure partant d'une solution initiale x_0.

2. ***Algorithmes Génétiques (AG)***

   Principe général : Il s'agit d'algorithmes d'exploration fondés sur les mécanismes de la sélection naturelle et de la génétique. Comme dans la nature, ils se fient à une part de hasard pour faire évoluer la solution. C'est la différence fondamentale avec les autres types d'algorithmes : il n'y a pas de déterminisme. A chaque génération, une nouvelle population (il faut définir cette notion pour chaque problème) est créée en conservant les meilleurs éléments (sélection) après avoir transformé (par mutation ou par croisement) l'ensemble de la population de la génération précédente. Un croisement est la combinaison de deux éléments pour en former un nouveau. Une mutation est le changement spontané d'un individu. La sélection assure que la solution progresse dans le bon sens mais la littérature sur le sujet prévient que la difficulté à surmonter est d'éviter que la population stagne dans le domaine d'attraction d'un minimum

local. En effet, la sélection peut éliminer des individus qui pourraient évoluer vers le minimum global car ils seront moins bons à ce moment que le reste de la population majoritairement dans le puits. Le rôle des mutations est justement de garantir que la sortie de tels domaines d'attraction reste toujours possible. La définition des mutations est donc importante dans l'implémentation de l'algorithme. En théorie, la probabilité des mutations doit être plus faible que celle des croisements.

Dans un AG, un individu (une solution) est caractérisé par son empreinte génétique. La force d'un individu peut être mesurée en se basant sur la valeur de la fonction objectif correspondante. La reproduction de l'évolution naturelle nécessite des opérateurs génétiques de croisement et de mutation qui sont des algorithmes agissant sur les chromosomes associés aux individus, et permettent de parcourir l'espace des solutions du problème [Goldberg, 1994].

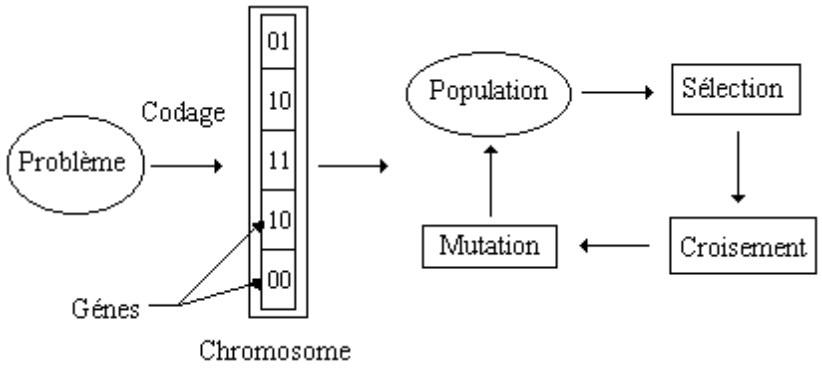

FIG. 1.6 – Principe des algorithmes génétiques

Les algorithmes génétiques, en tant que méthodes de recherche combinatoire, procurent un ensemble d'heuristiques efficaces de recherche dans des espaces complexes, sans requérir une connaissance approfondie du domaine considéré. Cette approche se montre donc particulièrement avantageuse pour traiter des problème complexes ("NP-complete problems", tels que le problème du voyageur de commerce), dont font partie les problèmes d'ordonnancement, pour lesquels de nombreuses techniques de recherche, essentiellement déterministes,

sont souvent mises en échec à cause des contraintes en temps de calcul. Cependant, l'application des algorithmes génétiques à ces problèmes soulève certaines difficultés : la représentation chromosomique (choix du codage) et l'évaluation des individus ne sont pas triviales ; souvent, des opérateurs génétiques, en plus des opérateurs traditionnels de croisement et de mutation, doivent être conçus pour donner des résultats utiles [Renders, 1995].

3. ***Recuit simulé***

Cette méthode d'optimisation a été proposée en 1982 par des spécialistes de physique statistique. Pour la recherche de la solution optimale, le recuit simulé procède à travers la génération de divers états d'énergie (chacun correspondant à une solution) commandés par la réduction de la température jusqu'à l'obtention du plus faible point d'énergie qui représente la meilleure solution. L'algorithme commence par générer la configuration initiale qui représente un point de l'espace des allocations possibles. A la configuration initiale sont associées une température et une énergie élevées. On définit le voisinage d'un point de l'espace comme l'ensemble des solutions qui peuvent être obtenues en déplaçant une tâche d'une ressource à une autre. L'algorithme procède par la génération et l'évaluation (grâce à une fonction coût) de configurations voisines. Cette politique d'exploration des solutions avoisinantes permet d'éviter le blocage dans des minima locaux [Baccouche, 1995].

Un algorithme d'optimisation par recuit simulé se décompose selon les étapes suivantes [Laquerbe et al., 1998] :

- ⋆ choix d'une fonction objectif à minimiser.
- ⋆ adoption d'un schéma de recuit dans lequel sont précisés la température initiale, le nombre de configurations générées à chaque température (nombre permettant d'atteindre un "état d'équilibre"), et le schéma de décroissance du paramètre de la méthode appelé toujours par analogie "température".
- ⋆ génération stochastique de configurations "voisines", correspondant aux transitions.
- ⋆ choix d'un critère d'acceptation de configurations dégradant le critère.

Le tableau 1.1 présente l'analogie entre la simulation thermodynamique et l'optimisation combinatoire.

TAB. 1.1 – Analogie entre la simulation thermodynamique et l'optimisation combinatoire

Simulation thermodynamique	**Optimisation combinatoire**
- État du système	- Solution admissible
- Énergie	- Coût
- Changement d'état	- Solution voisine
- Température	- Paramètre de contrôle
- État stable (gelé)	- Solution approchée

L'analogie entre les problèmes d'ordonnancement, le placement des tâches et le recuit simulé, est décrite dans le tableau 1.2.

TAB. 1.2 – Analogie entre les problèmes d'ordonnancement et le recuit simulé

Recuit simulé	**Ordonnancement**
Énergie	Fonction coût
État du système	Affectation : Opérations-Machines
Température	Paramètre de contrôle

4. ***Recherche Tabou***

La recherche tabou est une heuristique originalement développée par Glover [Glover, 1989, Glover, 1990]. Elle est basée sur des idées simples ; néanmoins elle est efficace. Cette méthode combine une procédure de recherche locale avec un certain nombre de règles et de mécanismes lui permettant de surmonter l'obstacle des extremums locaux, tout en les évitant. Elle a été appliquée avec succès pour résoudre de nombreux problèmes difficiles d'optimisation combinatoire : problèmes de routage de véhicules, problèmes d'affectation quadratique, problèmes d'ordonnancement, problèmes de coloration de graphes...

Principe de base : Dans une première phase, la méthode de recherche tabou peut être vue comme une généralisation des méthodes d'amélioration locale. En effet, en partant d'une solution quelconque x appartenant à l'ensemble de solutions X, on se déplace vers une solution $s(x)$ située dans le voisinage $[S(x)]$ de x. L'algorithme explore donc itérativement l'espace de solutions X.

Afin de choisir le meilleur voisin $s(x)$ dans $S(x)$, l'algorithme évalue la fonction objectif f en chaque point $s(x)$, et retient le voisin qui améliore la valeur de la fonction objectif f, ou au pire celui qui la dégrade le moins. L'originalité de la méthode de recherche tabou, par rapport aux autres méthodes locales, réside dans le fait que l'on retient le meilleur voisin, même si celui-ci est plus mauvais que la solution d'où l'on vient. Ce critère autorisant les dégradations de la fonction objectif évite à l'algorithme d'être piégé dans un minimum local, mais il induit un risque de répétitions cycliques. En effet, lorsque l'algorithme a quitté un minimum quelconque par acceptation de la dégradation de la fonction objectif, il peut revenir sur ses pas aux itérations suivantes.

Pour régler ce problème, l'algorithme a besoin d'une mémoire pour conserver pendant un moment la trace des dernières meilleures solutions déjà visitées. Ces solutions sont déclarées *taboues*. Elles sont stockées dans une liste de longueur L donnée, appelée *liste taboue*. Une nouvelle solution n'est acceptée que si elle n'appartient pas à cette liste taboue. Ce critère d'acceptation d'une nouvelle solution évite le bouclage de l'algorithme, durant la visite d'un nombre de solutions au moins égal à la longueur de la liste taboue, et dirige l'exploration de la méthode vers des régions du domaine de solutions non encore visitées.

5. ***Les colonies de fourmis*** C'est une nouvelle méthode de résolution des problèmes d'ordonnancement. Les colonies de fourmis sont basées sur le comportement réel de communication chez les fourmis qui consiste en « la trace » et « l'attrait ». Cette métaheuristique à été introduite la première fois dans la thèse de doctorat de Marco Dorigo (1992) et a été appliquée au problème du voyageur de commerce.

1.2.5.4 La logique floue et l'ordonnancement

La logique floue n'est pas vraiment une méthode de résolution mais représente un outil et concept performant pour prendre en considération l'ensemble des données et des paramètres incertains du problème abordé.

La logique floue présente, grâce au concept de fonction d'appartenance, un bon compromis entre la souplesse de l'emploi et la puissance de la représentation. Elle permet donc la modélisation de l'incertitude et de l'imprécision [Bouchon-Meunier, 1995]. Le moteur d'inférence a pour rôle de générer les solutions à partir de la base de connaissances et en respectant les règles.

1.3 Algorithmes génétiques et ordonnancement

1.3.1 Problème à une machine

Ce problème consiste à rechercher la séquence optimale de n tâches sur une seule machine. Une solution est décrite par l'ordre de passage des tâches sur la machine. On recense dans la littérature trois types de codages.

1. Le premier codage, proposé par Davis [Davis, 1985], est appelé "permutation". Il consiste simplement à ranger dans un vecteur le numéro des tâches dans l'ordre où on les place sur la machine.
2. Le deuxième codage est appelé "rang". Il consiste à donner à chaque tâche sa position relative sur la machine.
3. Le troisième codage, proposé par Portmann [Portmann, 1996], consiste a représenter le séquencement par une matrice de permutation.

1.3.2 Problème du flow-shop

La plupart des approches considèrent un codage de liste proposé par Syswarda [Syswarda, 1989]. Cette liste représentera l'ordre de passage des différents travaux

dans l'atelier.

1.3.3 Problème du job-shop

Plusieurs codages existent dans la littérature pour le job-shop classique.

- Le codage Yamada et Nakano [Yamada et Nakano, 1992] donne une représentation des dates réalisation de n des opérations pour chaque tâche. Concernant les opérateurs associés à ce codage, Yamada et al. ont proposé des algorithmes permettant de générer des ordonnancements actifs.
- Tamaki [Tamaki, 1992] utilise un codage binaire traduisant la représentation de la solution en graphe disjonctif. Les opérateurs génétiques classiques ne peuvent pas s'appliquer pour ce codage. D'autres types d'opérateurs mieux appropriés ont été développés [Tamaki, 1992].
- Kobayachi et al. [Kobayashi et al., 1995] représentent dans leur codage uniquement les séquences des opérations par machine. L'inconvénient de ce codage est la possibilité de violation des contraintes de précédence. Un processus de correction est indispensable après l'application d'un opérateur génétique.
- Portmann [Portmann, 1996] propose un codage indirect sous forme d'écriture matricielle de la présence d'un enchaînement entre deux opérations consécutives. Des opérateurs génétiques spécifiques à ce codage ont été également développés.

1.3.4 Problème du job-shop flexible

Pour le job shop flexible, Ghedjati [Ghedjati, 1994] propose un codage basé sur l'affectation avec un choix possible d'une heuristique pour chaque ressource. Ces heuristiques permettent alors à la ressource de choisir la tâche à réaliser. Ainsi, avec cette représentation un chromosome contient autant de gènes que de ressources.

Mesghouni [Mesghouni, 1999] propose aussi deux codages représentant des extensions des codages existants. Le premier utilise le codage de Kobayashi en ajoutant à chaque opération d'une séquence l'ordre de l'opération dans la gamme et sa date de début d'exécution. Le deuxième, une extension du codage de Yamada, introduit les

machines auxquelles sont affectées les opérations pour chaque tâche. L'inconvénient du premier codage est qu'il nécessite l'application d'un processus de correction. Les limites du deuxième codage sont ses difficultés d'intégrer le séquencement et d'appliquer des opérateurs agissant sur l'ordre de passage des opérations sur les machines. Kacem [Kacem, 2003], propose trois codages, un premier Codage Opérations-Machines (COM) qui donne les dates de début et de fin de chaque opération sur la machine à laquelle est affectée l'opération. Le deuxième codage est le Codage Liste des Opérations (CLO) qui représente l'ordonnancement dans un tableau de trois colonnes : opération, machine capable d'exécuter l'opération et dates de début et de fin. Le troisième codage est celui de Codage Liste des Séquencements des Jobs (CLSJ) représentant l'ordonnancement en n_j colonnes, n_j étant le nombre maximum d'opérations que peut contenir un job. Chaque colonne représente les tâches à ordonnancer sous forme d'une liste de x cellules, x étant le nombre de tâches. Chaque cellule est codée par le numéro de la tâche, la machine à laquelle l'opération est affectée et les dates de début et de fin de l'opération. La limite des deux premiers codages est qu'ils ne permettent pas une bonne exploration de l'espace de séquencement, et un bon usage des règles de priorités est nécessaire pour la détermination de l'ordre des passages sur les machines. Quand au troisième codage, il est difficile à implementer et nécessite beaucoup de temps de calcul.

1.4 Conception d'un système hybride d'aide à l'ordonnancement

Un système hybride d'aide à l'ordonnancement a été développé au sein de l'équipe «Ordonnancement complexes» du LAGIS de l'EC-Lille et du LAMIH de l'université de Valenciennes, dans le cadre d'un projet du GRAISyHM (Groupement de Recherche en Automatisation Intégrée et Système Homme/Machine). Les travaux développés ont pour objectif la conception d'un système hybride d'aide à l'ordonnancement constituant un environnement intégré permettant l'élaboration d'un ensemble d'ordonnancements admissibles, selon plusieurs responsables de la produc-

tion de formations hétérogènes et de sélectionner parmi cet ensemble ceux qui correspondent aux souhaits du responsable de l'atelier selon divers critères de performances (qualité des produits réalisés, encours raisonnables, retard minimal, charges des machines...).

Le système envisagé est caractérisé par [Liouane, 1998] :

– la coopération de nouvelles méthodes d'optimisation et d'apprentissage pour la résolution des problèmes d'ordonnancement : les algorithmes génétiques et leurs extensions, les algorithmes à stratégie d'évolution, les réseaux de Hopfield, la programmation sous contraintes, la logique floue...
– l'exploitation de plusieurs modèles de représentation des connaissances, permettant de tenir compte de connaissances moins structurées issues d'acteurs de la production de formations hétérogènes (ingénieurs en simulation, chefs de production, opérateurs,...), et ce par l'utilisation de la logique floue, l'approche multi-critères, les systèmes à base des connaissances,...

Le modèle conceptuel proposé par l'équipe est représenté par la figure 1.7.

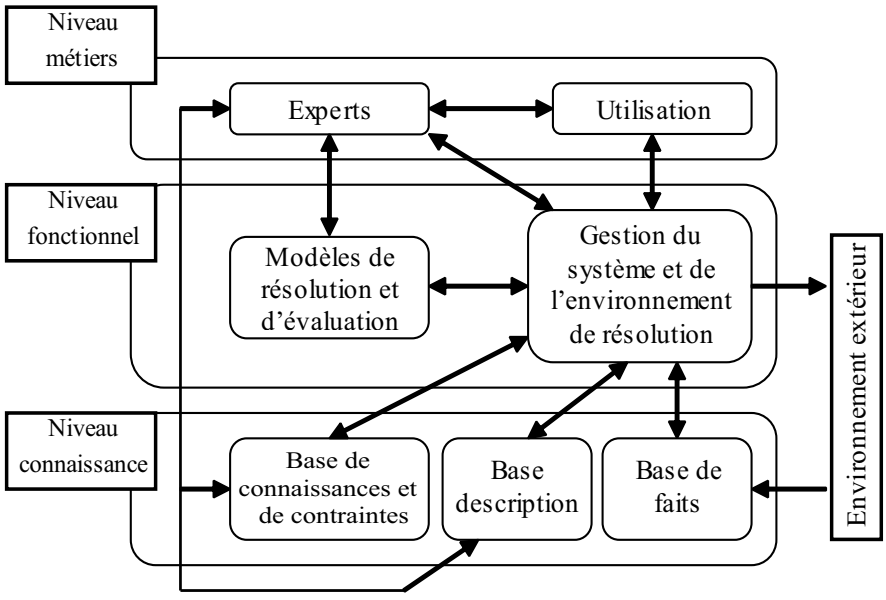

FIG. 1.7 – Architecture du système d'aide à l'ordonnancement

Le système d'aide à l'ordonnancement proposé par le projet du GRAISyHM fait

apparaître l'interaction entre le niveau métiers, le niveau fonctionnel et le niveau connaissances. La réussite de ces interactions peut être obtenue par la mise en œuvre d'un ensemble d'interfaces bien adaptées à chaque niveau du modèle. On distingue :

– les interfaces experts qui représentent l'ensemble des supports nécessaires aux experts pour bien mener et gérer les modules des niveaux fonctionnel et connaissance,
– les interfaces utilisateurs qui constituent l'outil de dialogue entre les opérateurs et les modules fonctionnels et assurent durant l'exploitation du système, la gestion des communications, tant au niveau informationnel (analyse, évaluation,...) qu'au niveau des actions exercées vers les modules fonctionnels.

1.4.1 Le niveau métiers

Le niveau métiers correspond aux divers intervenants : les experts qui participent à l'élaboration, la création et la gestion des bases de connaissances et au choix des modèles de pilotage, ainsi que les utilisateurs du système d'aide à l'ordonnancement [Liouane, 1998].

1.4.2 Le niveau fonctionnel

Ce niveau est constitué par deux modèles [Liouane, 1998].

- Un modèle de *résolution et d'évaluation* regroupe l'ensemble des modèles de résolution et d'exploitation de connaissances ainsi que la caractérisation et l'évaluation au sens des responsables de production, des solutions proposées. Il englobe plusieurs techniques de résolution des problèmes d'ordonnancement (partage des ressources, goulots d'étranglement, optimisation,...). Les techniques peuvent être de type analytique, heuristique, métaheuristique,...
- Un modèle *Gestion du système et de l'environnement de résolution* assure le contrôle et l'activation des mécanismes précédents, ainsi que des accès et l'exploitation des différentes bases du niveau connaissance. Il est construit autour d'un système interactif et un tableau de bord pour analyser la situation encours

et adopter la stratégie de pilotage proposée par les responsables de la production.

1.4.3 Le niveau connaissance

Ce niveau comprend :

- une base de *règles et de contraintes* spécifiques à l'application exprimant les modèles de connaissances "non procédurales" sur le domaine d'ordonnancement et le système applicatif,
- une base *description*, un modèle descriptif représentant les différentes données relatives aux éléments caractéristiques du domaine et le système applicatif,
- une base *de faits*, une base de données dynamique qui permet d'enregistrer les événements significatifs de l'évolution du processus de prise de décision (base des faits "décision") et du système applicatif (base des faits "application").

1.5 Choix des critères d'optimisation - Position du problème

Le job-shop flexible est une extension du problème classique de job-shop. Dans ce type de modèle, une opération nécessite une ressource pour être réalisée et cette ressource doit être choisie dans un ensemble défini a priori.

Ce type d'atelier flexible permet de modéliser de nombreux problèmes d'ordonnancement rencontrés en industrie mais aussi dans le secteur des services, où il est aussi souvent nécessaire de déterminer la bonne affectation pour chaque tâche. En effet, les ressources sont souvent des personnes et la flexibilité dans l'exécution des tâches vient d'une part du fait que plusieurs personnes de même qualification peuvent effectuer la même tâche, et d'autre part de la polyvalence des personnes qui peuvent faire différents types de tâches.

D'après l'état de l'art, dressé dans la section précédente, la prise en compte des critères sociaux-économiques dans les ateliers de production n'est que très récente, malgré sa pertinence au niveau industriel. Notons que la difficulté de ce type de

problèmes d'ordonnancement oblige le plus souvent les chercheurs à se limiter à des problèmes de tailles réduites et à des modèles basiques.

Il n'existe pas à notre connaissance d'étude concernant les systèmes incluant flexibilité et prise en compte des critères sociaux-économiques. Notre étude est motivée par la volonté de répondre en partie à ce manque. Dans les chapitres suivants, nous proposons d'aborder l'optimisation des systèmes de production des ateliers de type job-shop flexible en tenant compte des impératifs économiques.

1.6 Conclusion

Dans ce chapitre, nous avons survolé les principales caractéristiques d'un problème d'ordonnancement. Nous nous sommes intéressés ensuite à la complexité des problèmes qui peuvent être rencontrés et aux différentes méthodes de résolution, exactes et approchées envisageables.

Nous avons également présenté les données de base qui caractérisent un problème d'ordonnancement ainsi que les méthodes et les approches de résolution les plus courantes. Ces approches bien quelles soient assez nombreuses et diversifiées, comme il a été précédemment évoqué, demeurent toujours insuffisantes et peu adaptées à tous les types de problèmes d'ordonnancement.

Cependant, il faut noter que toutes les approches présentées se basent sur des informations collectées sur l'état du système, du processus de fabrication, des caractéristiques des opérations à exécuter et des données commerciales telles que : la configuration de l'atelier, les performances des ressources utilisées, les durées opératoires, les dates de disponibilité, les dates échues (dates de livraison clients), les retards, etc. Les caractéristiques des produits manipulés (matières premières, produits semi-finis et produits finis) n'ont pas été prises en considération. Ceci pourra représenter la limite de certaines approches. En effet, dans certains ateliers, les produits utilisés et fabriqués présentent des particularités qui doivent être considérées pour mener à bien la résolution des problèmes d'ordonnancement associés, par exemple dans le domaine agro-alimentaires.

Dans la suite, nous allons formuler les critères à minimiser, en appliquant une méthode de transformation statique des problèmes multi-objectifs en problèmes mono-objectif tenant comptes des interactions entre les critères.

Chapitre 2

Optimisation multi-critères à base de l'intégrale de Choquet tenant compte du coût de la production

Sommaire

2.1 Introduction

Dans le contexte industriel d'aujourd'hui, l'entreprise doit faire preuve de vigilance pour assurer efficacement la commande et le pilotage de son système productif. Il est donc évident qu'elle ne peut se passer de moyens fiables pour ne pas s'écarter des objectifs qu'elle se fixe et des budgets qu'elle s'alloue. l'ordonnancement représente l'une des fonctions les plus importantes pour la performance globale de ce système. En effet, c'est à ce niveau que les décisions vont être prises pour la régulation, la maîtrise du temps et des réalisations mobilisant toutes les ressources humaines, matérielles et informationnelles. Par ailleurs, dans une entreprise, on peut rencontrer plusieurs types de problèmes d'ordonnancement, notamment l'ordonnancement du personnel, l'ordonnancement d'atelier, l'ordonnancement de tournées de véhicules, l'ordonnancement de projet, ...

A l'heure actuelle, les problème d'ordonnancement de taille industrielle défient les méthodes exactes qui garantissent l'optimalité en durée exponentielle. L'intérêt de l'industrie se trouve donc focalisé sur le développement de méthodes approchées. Naturellement, ces méthodes doivent avoir des bases théoriques solides permettant d'approcher ces problèmes fortement combinatoires avec une certaine confiance dans la qualité du résultat. De plus, les besoins en réactivité et en flexibilité ont conduit à délaisser les méthodes qui rendent le système de production très sensible à la moindre variation des paramètres du modèle ainsi que de la nature et du nombre de critères à optimiser.

En effet, l'optimisation multi-objectifs cherche à optimiser plusieurs composantes d'un vecteur de fonctions objectif. Contrairement au mono-objectif, le problème multi-objectifs n'a pas une solution unique, mais un ensemble de solutions. Toute solution de cet ensemble est optimale dans le sens qu'aucune amélioration ne peut être faite sur une composante sans dégradation d'au moins une autre composante du vecteur [Talbi, 1999]. Compte tenu qu'une solution choisie par un décideur peut ne pas être acceptable par un autre, il s'avère utile de prévoir plusieurs alternatives au choix d'une solution optimale [Zitzler et Thiele, 1999].

Pour ce faire, une méthode d'optimisation multi-critères est proposée. Elle utilise l'agrégation par l'intégrale de Choquet. L'approche adoptée consiste à générer une variété de solutions optimales diversifiées dans l'espace de recherche de solutions, et à aider le décideur quand il ne peut pas donner une préférence particulière à l'une des fonctions objectif. D'une manière générale, les critères considérés présentent des relations non linéaires et complexes entre eux et n'ont pas forcement la même importance du point de vue du décideur. Ainsi, beaucoup de considérations peuvent être retenues pour tenir compte de toutes ces difficultés.

Dans ce chapitre, sont traités les problèmes d'ordonnancement dans les ateliers de production de type job-shop flexible où l'objectif principal est de produire le maximum de produits avec un temps d'exécution et un coût minimal. L'objectif est donc de rechercher un ordonnancement réalisable minimisant le Makespan, la charge critique, la charge totale, la pénalité avance/retard et le coût de fabrication, en appliquant des méthodes de transformation des problèmes multi-objectifs en problèmes mono-objectif [Collette et Siarry, 2002].

2.2 Optimisation multi-critères

Résoudre un problème d'optimisation consiste à trouver la ou les meilleure(s) solution(s), vérifiant un ensemble de contraintes et d'objectifs définis par l'utilisateur. Pour déterminer si une solution est meilleure qu'une autre, il est nécessaire que le problème introduise un critère de comparaison [Collette et Siarry, 2002].

2.2.1 Définitions

Définition 2.1 ***La fonction objectif***

C'est la fonction, notée f, qu'il s'agit d'optimiser.

Définition 2.2 ***Les variables de décision***

Elles sont regroupées dans un vecteur x. C'est en faisant varier ce vecteur que l'on recherche un optimum de la fonction objectif f.

Définition 2.3 ***Types de minima***

Il existe deux types de minima : les minima locaux et les minima globaux.

– *Minimum global : Un point x^* est un minimum global de la fonction f si on a :*

$$f(x^*) \leq f(x)\ \forall x \tag{2.1}$$

L'égalité n'étant vérifiée que pour $x^* = x$ si le minimum global est unique.

– *Minimum local : Un point x^* est un minimum local de la fonction f si on a :*

$$f(x^*) < f(x)\ \forall x \in V(x^*) \tag{2.2}$$

tel que $x^* \neq x$ et $V(x^*)$ définit un voisinage de x^*.

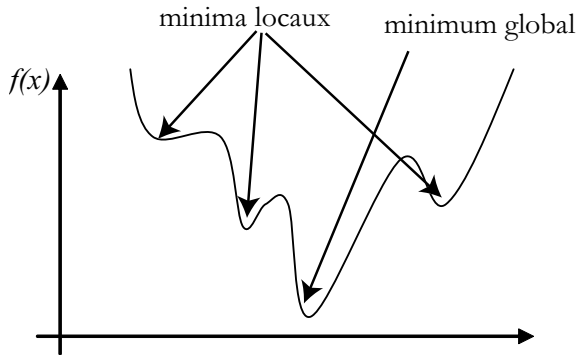

FIG. 2.1 – Les deux types de minima

2.2.2 Problèmes mono-objectifs

D'un point de vue mathématique, un problème d'optimisation mono-objectif est formulé de la façon suivante :

$$\begin{cases} \min f(x) & x \in \Re^n,\ f(x) \in \Re \\ g(x) \leq 0 & g(x) \in \Re^m \\ h(x) = 0 & h(x) \in \Re^l \end{cases} \tag{2.3}$$

avec m contraintes d'inégalités et l contraintes d'égalités.

2.2.3 Problèmes multi-objectifs

Lorsqu'on modélise un problème, il est souvent nécessaire de satisfaire plusieurs objectifs. Dans ce cas, on parle d'optimisation multi-objectifs [Tangour et Saad, 2006].

D'un point de vue mathématique, un problème d'optimisation multi-objectifs, se présente, dans le cas où le vecteur f regroupe k fonctions objectif, de la façon suivante :

$$\begin{cases} \min f(x) & x \in \Re^n,\ f(x) \in \Re^k \\ g(x) \leq 0 & g(x) \in \Re^m \\ h(x) = 0 & h(x) \in \Re^l \end{cases} \tag{2.4}$$

Comme il n'existe pas de relation d'ordre total dans un espace vectoriel, il est évident que cette approche nécessite une interprétation.

2.2.4 Notion de dominance

En ordonnancement, le concept de dominance d'un sous-ensemble de solutions est important afin de limiter la complexité algorithmique liée à la recherche d'une solution optimale au sein d'un grand ensemble de solutions.

Pour trouver une solution optimale aux problèmes d'optimisation multi-objectifs, constituant un ensemble de points, il s'avère nécessaire de définir une relation d'ordre entre ces éléments dite relation de dominance, pour identifier les meilleurs compromis. La règle de dominance est une contrainte qui peut être ajoutée au problème initial sans changer la valeur de l'optimum. La plus utilisée est celle définie au " sens de Paréto ".

Ainsi, la résolution d'un problème d'optimisation multi-objectifs conduit généralement à une multitude de solutions. Seul un nombre restreint de ces solutions est intéressant. Une solution est considérée intéressante s'il existe une relation de dominance entre cette solution et les autres solutions.

On dit que le vecteur x_1 domine le vecteur x_2 si :

- x_1 est au moins aussi bon que x_2 dans tous les objectifs,
- x_1 est strictement meilleur que x_2 dans au moins un objectif.

Les solutions qui dominent les autres mais ne se dominent pas entre elles sont appelées solutions optimales au sens de Paréto.

2.3 Méthodes d'agrégation des critères

Pour l'évaluation globale, deux approches agrégatives sont proposées : la première est l'agrégation par l'intégrale de Choquet et la seconde est l'agrégation par les opérateurs d'agrégation OWA (Ordered Weighted Averaging Operators) qui réalisent une sommation pondérée des divers critères.

2.3.1 Agrégation par l'intégrale de Choquet

L'opérateur d'agrégation à utiliser doit tenir compte de l'importance relative de chaque critère mais aussi de l'interaction entre les critères. Les intégrales floues en général et celle de Choquet en particulier permettent de représenter parfaitement ce genre de comportements. Dans ce qui suit, nous définissons cet opérateur ainsi que sa formulation mathématique [Grabisch, 1996].

Définition 2.4 *Une mesure floue sur* $N_c = 1, .., n_c$ *(ensemble de critères) est une fonction* $\mu : P(N_c) \rightarrow [0, 1]$*, vérifiant les axiomes suivants :*

1. $\mu(\emptyset) = 0$
2. $A \subset B \subset N_c \Rightarrow \mu(A) \leq \mu(B)$

$\mu(A)$ *représente l'importance ou le pouvoir de la coalition (groupe de critères)* A *pour le problème d'agrégation en question.*

Définition 2.5 *Soit* μ *une mesure floue sur* N_c*. L'intégrale de Choquet* C_μ *de* $a = (a_1, ..., a_{n_c})$*, vecteur des critères, par rapport à* μ *est définie par :*

$$C_\mu(a_1, \ ...a_{n_c}) \ = \ \sum_{i=1}^{n_c} (a_i - a_{i-1}) \mu(\{i, \ ...., n_c\}) \tag{2.5}$$

avec $a_0 = 0$ *et* $a_1 \leq ... \leq a_{n_c}$*.*

L'intégrale de Choquet est idempotente, continue, monotone et stable pour le changement d'échelle linéaire.

Nous rappelons deux notions qui sont importantes pour une analyse sémantique de l'intégrale de Choquet et des opérateurs couverts par celle-ci. Ce sont l'importance globale d'un critère et l'interaction entre critères.

– **Importance globale d'un critère**. *Il peut se définir par analogie avec la théorie des jeux coopératifs, où le joueur tient le rôle du critère. Nous avons vu que par définition $\mu(\{i\})$ traduit l'importance du critère C_i. Cependant, il se peut que cette valeur soit quasiment nulle, et pourtant que chaque fois que C_i se joint à une coalition $A \subset N_c$, l'importance de cette coalition s'en trouve sensiblement augmentée, ce qui tend à vouloir dire que C_i est un critère important. Cette dernière est l'indice de Shapley, ou valeur de Shapley issue de la théorie des jeux coopératifs. Pour tout critère C_i, son indice de Shapley est défini par :*

$$I_i = \sum_{K \subset N_c - \{i\}} \frac{(n_c - |K| - 1)!\,|K|!}{n_c!} \left(\mu(K \cup \{i\}) - \mu(K)\right) \tag{2.6}$$

où $|K|$ indique le cardinal de K et $0! = 1$.

– **Interaction entre deux critères**. *On peut pour deux critères i, j imaginer les trois situations suivantes.*

1. *L'importance de i et j pris ensemble, soit $\mu(\{i,j\})$, est plus grande que la somme des importances individuelles $\mu(\{i\}) + \mu(\{j\})$. Dans ce cas il y a une synergie de complémentarité entre ces deux critères.*

2. *$\mu(\{i,j\})$ est plus petite que la somme des importances individuelles. Cela signifie qu'il y a redondance ou une synergie négative entre ces deux critères.*

3. *$\mu(\{i,j\}) = \mu(\{i\}) + \mu(\{j\})$. Dans ce cas, on dit que les critères sont indépendants. En considérant comme importance globale toutes les coalitions possibles, on obtient la définition suivante pour l'indice d'interaction :*

$$I_{i,j} = \sum_{K \subset N_c - \{i,j\}} \frac{(n_c - |K| - 2)!\,|K|!}{(n_c - 1)!} (\mu(K \cup \{i,j\}) - \mu(K \cup \{i\}) - \mu(K \cup \{j\}) + \mu(K)) \tag{2.7}$$

L'expression de l'intégrale de Choquet en fonction des indices de Shapley et d'interaction est possible pour une mesure floue 2-additive :

$$C_\mu(a) = \sum_{I_{i,j} \succ 0} (a_i \wedge a_j) I_{i,j} + \sum_{I_{i,j} \prec 0} (a_i \vee a_j)\,|I_{i,j}| + \sum_{i=1}^{n_c} a_i (I_i - \frac{1}{2} \sum_{j \neq i} |I_{i,j}|) \tag{2.8}$$

où $\wedge$ et $\vee$ sont respectivement l'opérateur min et l'opérateur max.

2.3.2 Agrégation par les opérateurs OWA

Les opérateurs d'agrégation OWA (Ordered Weighted Averaging Operators) ont été introduits par Yager [Yager, 1988].

L'évaluation proposée consiste à transformer le problème multi-objectifs en un problème mono-objectif de la forme suivante :

$$C_{OWA}(x) = \sum_{i=1}^{n_c} w_i * a_i \tag{2.9}$$

où les poids w_i sont tels que $w_i \in [0,\ 1]$ et $\sum_{i=1}^{n_c} w_i = 1$. Ils expriment les préférences du décideur sur les critères.

Les opérateurs OWA permettent dans le cadre de l'agrégation multicritère de représenter principalement les deux concepts suivants [Kelman, 1996].

- Le décideur peut agir sur l'agrégation en imposant une satisfaction plus ou moins stricte de l'ensemble des critères : on peut représenter toutes les nuances allant de "tous les critères doivent être satisfaits" à "au moins un critère doit être satisfait" et cela intuitivement grâce à la liaison avec les quantifications linguistiques.
- Il est possible d'introduire une notion de pondérations entre les critères (OWA pondérés) : plus un critère est important dans la prise de décision finale, plus son poids est élevé (pondération statique).

La prise en compte simultanée de ces deux concepts par un seul opérateur est assez difficile à réaliser, car ces deux concepts sont très différents bien que tous deux très intuitifs.

2.3.3 Relation entre l'intégrale de Choquet et l'opérateur OWA

2.3.3.1 Exemple d'illustration de l'intégrale de Choquet

Le directeur d'une école d'ingénieurs veut évaluer ses étudiants à partir de leurs notes en Mathématiques (M), Statistiques (S) et Langues (L). Comme cette école est de type scientifique, le directeur ne veut pas admettre un étudiant faible dans les deux premières matières (M, S), et il désire si possible des étudiants également bons en langues. Ainsi, le directeur a décidé d'attribuer le coefficient 3 aux M, 3 aux S et 2 aux L. La somme pondérée C_{OWA} donne les résultats suivants pour les trois étudiants A, B et C (notes données sur une échelle de 0 à 20), table 2.1.

Tab. 2.1 – Evaluation globale par l'opérateur OWA et l'intégrale de Choquet

	M	S	L	C_{OWA}	C_μ
Étudiant A	18	16	10	15.25	13.9
Étudiant B	10	12	18	12.75	13.6
Étudiant C	14	15	15	14.62	14.9

Le directeur n'est du tout satisfait du résultat, car d'après lui, l'étudiant C est bon en sciences comme en littérature, et est meilleur que l'étudiant A, qui est excellent en maths et statistiques mais mauvais en langues. Alors, le directeur tente de changer les coefficients des matières, mais sans succès : attribuant les mêmes coefficients aux trois matières, A et C auront les mêmes scores et par conséquent le même rang (ce qui ne lui plaît pas). Compte tenu qu'il ne peut pas donner un coefficient plus important aux langues, il n'a donc pas de solution. Comment peut-on l'aider ? Simplement par l'utilisation des intégrales floues. En effet, les préférences de ce directeur peuvent

être formalisées par les trois règles suivantes :

1. les matières scientifiques (M, S) sont plus importantes ;
2. les matières scientifiques sont plus ou moins similaires, et les étudiants bons en mathématiques (respectivement statistiques) sont en général aussi bons en statistiques (respectivement mathématiques), et donc les étudiants qui sont bons en ces deux matières ne doivent pas être beaucoup favorisés ;
3. les étudiants bons en maths (resp. S) et en littérature sont très rares et doivent être favorisés.

Ce cas est un exemple typique de l'interaction entre les critères. L'opérateur d'agrégation à utiliser doit tenir compte de l'importance de chaque critère pris à part mais aussi de l'interaction entre les critères. Les intégrales floues, en général, et celle de Choquet, en particulier, permettent de représenter ce genre de comportements parfaitement. Nous verrons par la suite comment on peut aider ce directeur. Revenons maintenant à notre exemple introductif, et voyons comment les préférences du directeur peuvent être traduites en termes de mesures floues :

1. $\mu(\{M\}) = \mu(\{S\}) = 0.45$, $\mu(\{L\}) = 0.3$ (Importances relatives des critères)
2. $\mu(\{M, S\}) = 0.5 \prec \mu(\{M\}) + \mu(\{S\})$ (Redondance)
3. $\mu(\{M, L\}) = \mu(\{S, L\}) = 0.9 \succ 0.45 + 0.3$ (Complémentarité)

L'application de l'intégrale de Choquet C_μ avec les mesures floues ci-dessus donne le résultat du la table 2.1.

Détaillons le calcul de C_μ pour l'étudiant A

Tout d'abord, classons les notes de l'étudiant A dans l'ordre croissant pour pouvoir appliquer l'equation (2.5) :

$$10 \leq 16 \leq 18$$

Il vient le classement des matières suivant :

$$\{L, S, M\}$$

Appliquons l'intégrale de Choquet selon l'équation (2.5) :

$$C_\mu(A) = (10-0)\mu(\{M,S,L\}) + (16-10)\mu(\{M,S\}) + (18-16)\mu(\{M\})$$

Il vient :

$$C_\mu(A) = 10 + 6 * 0.5 + 2 * 0.45 = 13.9$$

De même pour l'étudiant B

$$10 \leq 12 \leq 18 \qquad \text{implique} \qquad \{M,\ S,\ L\}$$

L'application de l'intégrale de Choquet selon l'équation (2.5) donne :

$$C_\mu(B) = (10-0)\mu(\{M,S,L\}) + (12-10)\mu(\{L,S\}) + (18-12)\mu(\{L\})$$

ou encore :

$$C_\mu(B) = 10 + 2 * 0.9 + 6 * 0.3 = 13.6$$

De même pour l'étudiant C

$$14 \leq 15 \leq 15 \qquad \text{implique} \qquad \{M,\ S,\ L\}$$

l'application de l'intégrale de Choquet selon l'équation (2.5) donne :

$$C_\mu(B) = (14-0)\mu(\{M,S,L\}) + (15-14)\mu(\{L,S\}) + (15-15)\mu(\{L\})$$

Il vient :

$$C_\mu(B) = 14 + 1 * 0.9 + 0 = 14.9$$

Les étudiants ont été ainsi ordonnés en concordance avec les préférences du directeur. Nous signalons que l'étudiant B est toujours en dernier rang, comme voulait la tendance scientifique de cette école.

2.3.3.2 Particularité de l'intégrale de Choquet

La stratégie de décideur adoptée est modélisée par la matrice d'interaction de la relation (2.10).

$$I = \begin{pmatrix} w_1 & I_{12} & I_{13} & I_{14} & I_{15} \\ I_{12} & w_2 & I_{23} & I_{24} & I_{25} \\ I_{13} & I_{23} & w_3 & I_{34} & I_{35} \\ I_{14} & I_{24} & I_{34} & w_4 & I_{45} \\ I_{15} & I_{25} & I_{35} & I_{45} & w_5 \end{pmatrix} \tag{2.10}$$

$$C_\mu(x) = \sum_{I_{i,j} \succ 0} (a_i \wedge a_j) I_{i,j} + \sum_{I_{i,j} \prec 0} (a_i \vee a_j) \, |I_{i,j}| \; + \sum_{i=1}^{n_c} a_i (I_i - \frac{1}{2} \sum_{j \neq i} |I_{i,j}|) \tag{2.11}$$

Avec $I_i = w_i$ et $C_{OWA}(x) = \sum_{i=1}^{n_c} w_i \, a_i$, il vient : $C_\mu(x)$ peut être réécrit sous la forme suivante :

$$C_\mu(x) = \sum_{I_{i,j} \succ 0} (a_i \wedge a_j) I_{i,j} + \sum_{I_{i,j} \prec 0} (a_i \vee a_j) \, |I_{i,j}| \; - \frac{1}{2} \sum_{i=1}^{n_c} \sum_{j \neq i} |I_{i,j}| + C_{OWA}(x) \tag{2.12}$$

L'opérateur OWA est donc un cas particulier de l'intégrale de Choquet, relatif au cas où l'interaction entre les critères est nulle : $I_{i,j} = 0 \; \forall i \neq j$

2.4 Formulation du problème job-shop flexible

Le problème d'ordonnancement du job-shop flexible présente la difficulté de l'affectation des tâches de chaque produit à une machine, et celle relative à l'ordonnancement de l'ensemble des tâches dans un ordre minimisant un certain critère.

Le but de l'étude envisagée est de rechercher un ordonnancement réalisable qui minimise le Makespan, la charge critique, la charge totale, la pénalité avance/retard ainsi que le coût de fabrication.

Les données du problème considéré sont les suivantes :

- il y a N produits indépendants, soit Jj, $j = 1, 2, ..., N$;
- chaque produit J_j nécessite un ensemble d'opérations à réaliser selon un ordre bien défini, spécifique à la gamme de fabrication G_j ;
- chaque gamme de fabrication G_j nécessite une série de n_j opérations $O_{i,j}$, $i = 1, ..., n_j$;
- la réalisation de chaque opération $O_{i,j}$ nécessite la sélection d'une ressource ou d'une machine M_k parmi M machines, $k = 1, ..., M$;
- chaque machine ne peut réaliser qu'une seule opération à la fois ;
- la préemption n'est pas autorisée.

Notations :

r_j : date de début au plus tôt du produit J_j

d_j : date de fin au plus tard à laquelle on désire avoir terminé l'exécution du produit J_j

tf_j : date de fin d'exécution du produit J_j

$tf_{i,j,k}$: date de fin d'exécution de l'opération $O_{i,j}$ sur la machine M_k

$r_{i,j,k}$: date de début d'exécution de l'opération $O_{i,j}$ sur la machine M_k

$p_{i,j,k}$: durée d'exécution de l'opération $O_{i,j}$ sur la machine M_k

$\alpha_{i,j}$: durée minimale de $O_{i,j}$, $\alpha_{i,j} = \min_{1 \leq k \leq M}(d_{i,j,k})$,

w_k : charge de la machine M_k

$\omega_{j,k}$: coefficient d'utilisation de la machine M_k pour la production du produit J_j, $\omega_{j,k} \in \{0, 1\}$

w_{kj} : charge de la machine M_k pour le produit J_j

Mp_j : matière première du produit J_j

Cm_k : coût unitaire sur la machine M_k

Cf_j : coût de fabrication du produit J_j

Les solutions sont représentées sous la forme du Codage Parallèle d'Ordres de Fabrication "CPOF", proposé par [Mesghouni, 1999]. Une solution est représentée par une matrice (Tab.2.2), chaque ligne représente la gamme de chaque ordre de fabrication. Chaque cellule de cette ligne, représentant une opération $O_{i,j}$, contient trois termes.

Le premier indique le numéro de la machines M_k qui est affectée à l'exécution de cette opération, le second représente la date de début d'exécution $r_{i,j,k}$ de l'opération et le troisième représente la date de fin d'exécution $tf_{i,j,k}$ de l'opération, dates calculées en tenant compte des contraintes de ressource et de précédence.

TAB. 2.2 – Codage CPOF

	O_1	O_2	O_i
P1	M_k ; $r_{1,1,k}$; $tf_{1,1,k}$	M_k ; $r_{2,1,k}$; $tf_{2,1,k}$	...
Pj	M_k ; $r_{1,j,k}$; $tf_{1,j,k}$	...	M_k ; $r_{i,j,k}$; $tf_{i,j,k}$

2.4.1 Formulation des critères et du coût total de production

Les critères considérés sont au nombre de cinq, dont les trois premiers constituent des critères classiques utilisés pour l'optimisation des problèmes d'ordonnancement de type job-shop flexible [Kacem et al., 2002][Xia et Wu, 2005], et les deux derniers sont deux critères économiques récemment élaborés [Saad et al., 2007b].

Les objectifs considérés sont relatifs à la minimisation :

– du Makespan C_1 :

$$C_1 = C_{\max} = \max_{1 \leq j \leq N} (tf_j) \tag{2.13}$$

– de la charge critique C_2 :

$$C_2 = \max_{1 \leq k \leq M} (w_k) \tag{2.14}$$

– de la charge totale des machines C_3 :

$$C_3 = \sum_{k=1}^{M} w_k \tag{2.15}$$

– de la pénalité d'avance (pénalité de stockage) et de retard C_4 :

$$C_4 = \sum_{j=1}^{N} A_j = \sum_{j=1}^{N} (h_j E_j + b_j T_j) \tag{2.16}$$

avec pour l'avance :

$$E_j = \max(0,\, d_j - tf_j) \tag{2.17}$$

et pour le retard :

$$T_j = \max(0, tf_j - d_j) \tag{2.18}$$

h_j et b_j étant les coûts de pénalisation du produit J_j respectivement pour l'avance et le retard, et A_j la pénalité pour le produit J_j,

– du coût de fabrication des produits C_5 :

$$C_5 = \sum_{1\leq j\leq N} Cf_j = \sum_{1\leq j\leq N} \left(Mp_j + \sum_{1\leq i\leq n_j} \left(\sum_{1\leq k\leq M} (\omega_{j,k}\, Cm_k\, p_{i,j,k}) \right) \right) \tag{2.19}$$

Le coût de la production d'un produit J_j étant Cp_j [Saad et al., 2006] :

$$Cp_j = A_j + Cf_j \tag{2.20}$$

le coût total de production C_p est donc :

$$Cp = C_4 + C_5 \tag{2.21}$$

2.4.2 Formulation des bornes inférieures

2.4.2.1 Bornes du Makespan

En supposant qu'il n'y pas d'intervalle d'attente entre deux opérations successives, considérons la minoration suivante :

$$r_j + \sum_{i=1}^{n_j} \alpha_{i,j} \leq tf_j, \qquad \forall j = 1, ..., N \tag{2.22}$$

Par application de la fonction " max " sur l'inégalité (2.22), il vient :

$$\max_{1\leq j\leq N} \left\{ r_j + \sum_{i=1}^{n_j} \alpha_{i,j} \right\} \leq \max_{1\leq j\leq N} (tf_j) \tag{2.23}$$

ou encore :

$$\max_{1\leq j\leq N}\left\{r_j+\sum_{i=1}^{n_j}\alpha_{i,j}\right\}\leq C_{\max} \tag{2.24}$$

Le premier membre de cette inégalité constitue une borne inférieure du Makespan des produits :

$$C_{b1}=\max_{1\leq j\leq N}\left\{r_j+\sum_{i=1}^{n_j}\alpha_{i,j}\right\} \tag{2.25}$$

Si le cardinal de l'ensemble des N produits est supérieur au nombre de machines, ou dans le cas de relaxation de certaines contraintes (préemption des tâches, contrainte disjonctive sur les ressources,...), il n'y a pas de solution atteignant la borne inférieure [Berkoune et al., 2005]. La méthode de calcul d'une borne inférieure possible C_{b2}, à appliquer dans ce cas, est la suivante :

$$C_{b2}=\frac{1}{M}\left(\sum_{k=1}^{M}r_k^{'}+\sum_{j=1}^{N}\sum_{i=1}^{n_j}\alpha_{i,j}\right) \tag{2.26}$$

où $r_k^{'}$ est la date de disponibilité au plus tôt de la machine k.
Les bornes C_{b1} et C_{b2} ainsi définies permettent de prévoir des limites pour les valeurs du Makespan :

$$C_1^b=\max\left\{C_{b1},C_{b2}\right\}\leq C_{\max} \tag{2.27}$$

2.4.2.2 Borne inférieure de la charge critique des machines

La borne inférieure de la charge critique des machines, notée C_2^b, est définie par [Kacem, 2003] :

$$C_2^b=E\left(\frac{\sum_{j=1}^{N}\left(\sum_{i=1}^{n_j}\alpha_{i,j}\right)}{M}\right) \tag{2.28}$$

où E(.) est la partie entière.

2.4.2.3 Borne inférieure de la charge totale des machines

La borne inférieure de la charge totale des machines, notée C_3^b, est définie par [Kacem et al., 2001] :

$$C_3^b = \sum_{j=1}^{N} \left(\sum_{i=1}^{n_j} \alpha_{i,j} \right) \tag{2.29}$$

2.4.2.4 Borne inférieure de la pénalité d'avance et du retard

En supposant que les produits finissent juste à temps (pas de stocks et pas de pénalités), la valeur du critère C_4 est nulle :

$$C_4 \geq 0 \tag{2.30}$$

La borne inférieure relative à la pénalité d'avance\retard est donc zéro [Saad et al., 2007b].

2.4.2.5 Borne inférieure du coût de fabrication des produits

En considérant que les produits finissent juste à temps (pas de stocks et pas de pénalités), l'inégalité suivante est toujours vraie pour chaque opération $O_{i,j}$:

$$Cm_k \; p_{i,j,k} \geq \min_k (Cm_k \; p_{i,j,k}) \tag{2.31}$$

La quantité suivante C_5^b constitue, dans ce cas, une borne inférieure du coût de fabrication des produits [Saad et Benrejeb, 2006] :

$$C_5^b = \sum_{j=1}^{N} Mp_j + \sum_{j=1}^{N} \sum_{i=1}^{n_j} \min_k (p_{i,j,k} \, . \, Cm_k) \tag{2.32}$$

2.5 Approche d'évaluation multi-critères à l'aide de l'intégrale de Choquet

Nous nous intéressons à évaluer et à comparer les solutions selon plusieurs critères. D'une manière générale, ces critères présentent des relations non linéaires et com-

plexes entre eux et n'ont pas forcément la même importance du point de vue du décideur. Ainsi, beaucoup de considérations peuvent être prises pour tenir compte de toutes ces difficultés [Sidi, 2006]. Pour ce faire, nous utilisons une méthode d'évaluation floue basée sur les étapes suivantes :

– pour chaque fonction objectif, une borne inférieure est calculée, telle que :

$$C_i(x) \geq C_i^b \ \ \forall x \in S, \quad 1 \leq i \leq n_c \tag{2.33}$$

où S est l'espace des solutions réalisables et n_c nombre de fonctions objectif ;

– les valeurs des fonctions objectif, dans la plupart des cas, peuvent appartenir à différents intervalles de magnitude variable.

En particulier, soit H une heuristique choisie et C_i^h la valeur maximale de la solution donnée par l'heuristique considérée selon la ième fonction objectif.

La fuzzification est appliquée en utilisant les fonctions d'appartenance décrites sur la figure (fig.2.2).

Un vecteur $C(x)$ est associé à chaque solution réalisable x, $C(x) \in \left[C_1^b, +\infty\right] \times ... \times \left[C_{n_c}^b, +\infty\right]$, avec $C(x) = (C_1(x), ..., C_{n_c}(x))^T$. Pour chaque vecteur $C(x)$, une fuzzification de ses composantes $C_i(x)$, proposée selon leurs positions dans les intervalles $\left[C_i^b, C_i^h\right]$, est considérée en deux sous-ensembles flous B^i et M^i, figure fig.2.2. On a :

$$\mu_i^B(C_i(x)) = \frac{\mathrm{C_i^h - C_i(x)+\varepsilon}}{C_i^h - C_i^b + \varepsilon}, \text{ si } C_i(x) \in \left[C_i^b,\ C_i^h + \varepsilon\right] \tag{2.34}$$

$$\mu_i^B(C_i(x)) = 0, \text{ sinon}$$

$\mu_i^B(C_i(x))$ étant la mesure floue de $C_i(x)$ dans le sous-ensemble B^i.

La qualité de chaque solution x est ensuite caractérisée dans (2.35) par le vecteur $C_B(x)$ dont toutes les composantes sont homogènes puisqu'elles appartiennent toutes au même intervalle $[0, 1]$ et sont toutes sans dimension :

$$\begin{aligned} &C_B(x) = (a_1, ..., a_{n_c})^T \\ &a_i = \mu_i^B(C_i(x)), \quad \forall i = 1, 2, ..., n_c \end{aligned} \tag{2.35}$$

Pour l'évaluation globale, l'évaluation des critères $C_g(x)$ de l'approche proposée est à base de l'intégrale de Choquet $1 - C_\mu(x)$.

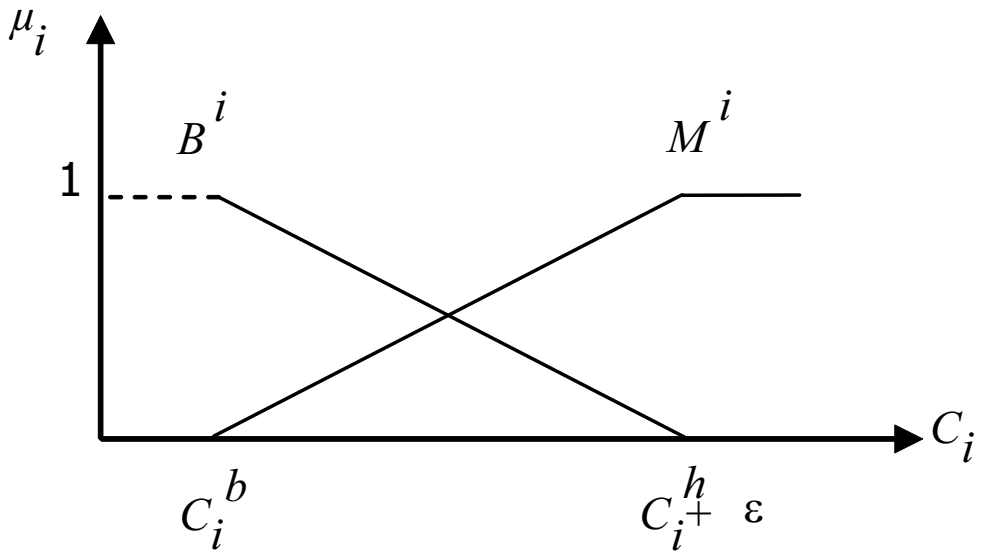

FIG. 2.2 – Application floue dans la résolution du problème d'échelle

2.5.1 Préférence de décideur

L'importance des critères est définie selon la stratégie du décideur. Cette importance peut être traduite comme suit :

– C_1 est le critère le plus important,

– C_3 et C_4 sont moins importants que C_1,

– C_2 et C_5 sont considérés négligeables.

Nous avons défini l'indice de Shapley comme le coefficient d'importance de critères. En effet, l'indice d'importance n'est pas suffisant pour avoir une bonne description du comportement des critères dans un problème de décision. Évidemment, l'indice de Shapley n'est pas réduit à la mesure floue parce que les critères agissent ensemble, ou bien ils ont intérêt de coopérer ensemble. Dans un processus décisionnel multi-critères, la situation est différente et la compréhension des phénomènes d'interaction entre les critères est plus informative parce que l'importance des critères n'est pas suffisante pour décrire un modèle de décision.

Après la génération des décisions, le module d'évaluation est appelé pour déterminer la meilleure décision réalisable en tenant compte de l'importance de chaque critère et

de l'interaction entre les critères. La stratégie de régulation adoptée par le régulateur doit avoir été modélisée par la matrice représentant les poids des critères et leurs interactions (coefficients de l'intégrale de Choquet) par exemple :

$$I = \begin{pmatrix} 0.4 & 0.5 & 0.1 & -0.5 & 0.4 \\ 0.5 & 0.1 & -0.4 & 0.3 & -0.1 \\ 0.1 & -0.4 & 0.2 & 0.5 & -0.2 \\ -0.5 & 0.3 & 0.5 & 0.2 & 0.3 \\ 0.4 & -0.1 & -0.2 & 0.3 & 0.1 \end{pmatrix} \tag{2.36}$$

2.5.2 Exemple d'application de l'intégrale de Choquet sur un problème 5 machines-3 produits

Considérons un système de production flexible composé de 5 machines, notées $M = \{M_k; \forall k = 1, .., 5\}$. Cet ensemble de machines fabrique 3 produits notés $P = \{J_j; \forall j = 1, 2, 3\}$. La table suivante (Tab.2.3) précise le nombre et l'ordre des opérations de chaque produit ainsi que les durées opératoires $p_{i,j,k}$ relatives à chaque machine.

Tab. 2.3 – Exemple : 3 produits - 5 machines

		M_1	M_2	M_3	M_4	M_5
J_1	$O_{1,1}$	1.5	9.5	3.12	4.91	4.5
	$O_{2,1}$	3	4.5	1.75	4.7	4.5
	$O_{3,1}$	4.5	7	1.75	4.5	3.75
J_2	$O_{1,2}$	1.5	4.5	4.5	3.25	6.37
	$O_{2,2}$	1.5	8.25	4.91	4.5	3.75
	$O_{3,2}$	4.5	4.5	1.75	2	4.5
J_3	$O_{1,3}$	1.5	4.5	4.91	3.25	3
	$O_{2,3}$	4.5	9.5	1.75	4.5	3.75

Avec l'ensemble des dates de début d'exécution au plus tôt $\{r_j\} = \{0;\ \forall\ j = 1, .., 3\}$ et l'ensemble des dates de fin d'exécution au plus tard de chaque produit

$\{d_1 = 7, d_2 = 7, d_3 = 6\}$, les coûts de pénalité et de matières premières sont les suivants (en unité monétaire) :

$$Mp_j = 2,\ h_j = 0.5 \text{ et } b_j = 2,\ j = 1, 2, ..., n$$
$$Cm_k = 3, k = 1, 2, ..., m$$

Soit la matrice d'interaction I :

$$I = \begin{pmatrix} 0.4 & 0.5 & 0.1 & -0.5 & 0.4 \\ 0.5 & 0.1 & -0.4 & 0.3 & -0.1 \\ 0.1 & -0.4 & 0.2 & 0.5 & -0.2 \\ -0.5 & 0.3 & 0.5 & 0.2 & 0.3 \\ 0.4 & -0.1 & -0.2 & 0.3 & 0.1 \end{pmatrix}$$

Quatre solutions possibles sont obtenues pour cet exemple, table 2.4. Ces solutions sont choisies de façon à ce qu'il respectent les données de l'exemple.

TAB. 2.4 – Les solutions du 1^{er} exemple 3 produits - 5 Machine

(a)	O_1	O_2	O_3
J_1	1 ;0 ;1.5	3 ;1.5 ;3.25	5 ;3.25 ;7
J_2	1 ;1.5 ;3	1 ;3 ;4.5	3 ;5 ;6.75
J_3	4 ;0 ;3.25	3 ;3.25 ;5	*****

(b)	O_1	O_2	O_3
J_1	1 ;0 ;1.5	3 ;1.5 ;3.25	3 ;6.25 ;8
J_2	1 ;1.5 ;3	5 ;3 ;6.75	4 ;6.75 ;8.75
J_3	1 ;3 ;4.5	3 ;4.5 ;6.25	*****

(c)	O_1	O_2	O_3
J_1	1 ;0 ;1.5	3 ;1.5 ;3.25	3 ;5 ;6.75
J_2	1 ;1.5 ;3	1 ;3 ;4.5	4 ;4.5 ;6.5
J_3	5 ;0 ;3	3 ;3.25 ;5	*****

(d)	O_1	O_2	O_3
J_1	1 ;0 ;1.5	3 ;1.5 ;3.25	3 ;5 ;6.75
J_2	4 ;0 ;3.25	1 ;3.25 ;4.75	4 ;4.75 ;6.75
J_3	1 ;1.5 ;3	3 ;3.25 ;5	*****

La table 2.5 montre les valeurs de critères pour chaque solution.

TAB. 2.5 – Valeurs des critères

	C_1	C_2	C_3	C_4	C_5
a	7	5,25	16,75	0,625	56,25
b	8,75	5,25	15,5	0,6	52,5
c	6,75	5,25	14,75	0,875	50,25
d	6,75	5,25	15	0,75	51

Par application de la méthode de normalisation des critères décrite au par avant, nous obtenons les valeurs de critères entre 0 et 1, $a_i = \mu_i^B(C_i(x))$. Ces valeurs ainsi que le score obtenu par l'intégrale de Choquet sont données dans la table tab.2.6.

TAB. 2.6 – Valeurs normalisées des critères pour chaque solution

	a_1	a_2	a_3	a_4	a_5	C_μ
a	0,6	0,25	0,779	0,938	0,662	0,604
b	0,75	0,25	0,897	0,913	0,779	0,330
c	0,75	0,25	0,882	0.925	0.765	0,697
d	0,75	0,25	0,882	0,925	0,765	0,685

Pour plus de détails, reprenons l'équation (2.8) de l'intégrale de choquet en la décomposant sur trois éléments :

$$C_\mu(a) = \underbrace{\sum_{I_{i,j} \succ 0} (a_i \wedge a_j) I_{i,j}}_{\text{z1}} + \underbrace{\sum_{I_{i,j} \prec 0} (a_i \vee a_j) \, |I_{i,j}|}_{\text{z2}} + \sum_{i=1}^{n_c} a_i (I_i - \frac{1}{2} \sum_{j \neq i} |I_{i,j}|)$$

Reformulons les éléments z1 et z2 de façon simple :

$$\text{z1} = \sum_{\substack{I_{i,j}>0 \\ i \neq j}} (a_i \wedge a_j) I_{i,j} = \sum_{\substack{I_{i,j}>0 \\ i \neq j}} Min(a_i, a_j) * I_{i,j} = \sum_i \sum_j Min(a_i, a_j) * I^1_{i,j}$$

$$\text{z2} = \sum_{\substack{I_{i,j}<0 \\ i \neq j}} (a_i \vee a_j) I_{i,j} = \sum_{\substack{I_{i,j}>0 \\ i \neq j}} Max(a_i, a_j) * I_{i,j} = \sum_i \sum_j Max(a_i, a_j) * I^2_{i,j}$$

sachant que :

$$I^1 = \begin{pmatrix} 0 & 0.5 & 0.1 & 0 & 0.4 \\ 0.5 & 0 & 0 & 0.3 & 0 \\ 0.1 & 0 & 0 & 0.5 & 0 \\ 0 & 0.3 & 0.5 & 0 & 0.3 \\ 0.4 & 0 & 0 & 0.3 & 0 \end{pmatrix} \quad I^2 = \begin{pmatrix} 0 & 0 & 0 & -0.5 & 0 \\ 0 & 0 & -0.4 & 0 & -0.1 \\ 0 & -0.4 & 0 & 0 & -0.2 \\ -0.5 & 0 & 0 & 0 & 0 \\ 0 & -0.1 & -0.2 & 0 & 0 \end{pmatrix}$$

D'après la table 2.6 et les équations ci-dessus, le score C_μ obtenu par Choquet intégral est facilement calculable. la solution ($\boldsymbol{c}$) admet le meilleur score obtenu par l'intégrale de Choquet. C'est aussi la solution obtenue par application de l'algorithme génétique à base de l'intégrale de Choquet pour la résolution des problèmes d'ordonnancement des ateliers flexibles de type job-shop.

2.6 Simulation de l'approche d'évaluation basée sur l'intégrale de Choquet

Quatre benchmarks de type job-shop flexible $N \times M$, traitant N produits sur M machines [Xia et Wu, 2005], sont traités ici pour montrer la validité de la mise en œuvre de la méthodes d'optimisation multi-critères proposée dans le paragraphe précédent. Les coûts de pénalité et le coût des matières premières sont, dans les deux cas, les suivants (en unité monétaire) :

$$Mp_j = 2,\ h_j = 0,5\ et\ b_j = 2,\ j = 1, 2, ..., N$$
$$Cm_k = 3,\ k = 1, 2, ..., M$$

les critères à optimiser au nombre de 5 étant formulés par les relations (2.13) à (2.16) et (2.19).

Dans cette section, les différents résultats obtenus par l'utilisation de la méthode d'optimisation multi-critères par l'intégrale de Choquet sont présentés dans les tables 2.9, 2.14, 2.15 et 2.16. Ces résultats sont comparés avec ceux obtenus par d'autres

TAB. 2.7 – Benchmarks étudier par l'approche d'évaluation basée sur l'intégrale de Choquet

$N \times M$	durées opératoires	solutions	évaluation
8×8	table A.3	table 2.9	table 2.10
10×10	table A.4	table 2.14	table 2.11
10×7	table A.5	table 2.15	table 2.12
15×10	table A.6	table 2.16	table 2.13

méthodes dans la table 2.8 : la colonne notée 'AL' est relative à l'approche par localisation [Kacem et al., 2002] et la colonne suivante 'PSO+SA' est relative à l'optimisation par recuit simulé [Xia et Wu, 2005]. Les Valeurs des critères relatives aux différents benchmarks sont présentées dans les tables 2.10, 2.11, 2.12 et 2.13.

TAB. 2.8 – Comparaison des résultats obtenus pour les benchmarks étudiés

	Bornes inférieures			AL			PSO+SA			Intégrale de Choquet		
$N \times M$	C_1^b	C_2^b	C_3^b	C_1	C_2	C_3	C_1	C_2	C_3	C_1	C_2	C_3
8×8	12	10	73	16	13	75	16	13	73	16	13	75
10×10	7	5	41	7	5	45	7	6	44	7	5	44
10×7	11	9	60	15	11	61				11	10	64
15×10	23	10	91	23	11	95				23	11	99

Les différents résultats montrent que les solutions obtenues sont généralement acceptables et satisfaisantes, malgré le fait que cinq critères sont considérés au lieu de trois critères dans [Kacem et al., 2002] et [Xia et Wu, 2005]. Les valeurs des fonctions objectifs montrent l'efficacité de l'approche proposée (table 2.8). Pour chaque benchmark, cette méthode a permis d'obtenir une variante de solutions, qui ont les mêmes valeurs de critères et a donné des solutions meilleures pour les benchmarks 10×10 et 10×7.

L'approche proposée est basée sur une hybridation des algorithmes évolutionnistes et la logique floue. La méthode mise en oeuvre a permis d'obtenir des résultats

satisfaisants. L'efficacité de cette approche est prouvée par le choix judicieux des indices d'interaction des critères et par la qualité de l'intégrale de Choquet pour l'agrégation. En outre, une amélioration nette de la qualité des solutions obtenues est due, sans aucun doute, à l'utilisation de l'évaluation floue, et à la méthode agrégative utilisée pour l'optimisation des solutions.

TAB. 2.9 – Solutions relatives au benchmark 8×8

	O_1	O_2	O_3	O_4
J_1	2 ;0 ;3	5 ;3 ;6	6 ;9 ;11	
J_2	3 ;0 ;3	4 ;3 ;5	7 ;9 ;10	5 ;10 ;14
J_3	7 ;0 ;2	4 ;5 ;9	1 ;9 ;10	
J_4	2 ;3 ;4	6 ;4 ;9	3 ;9 ;11	
J_5	1 ;0 ;3	7 ;3 ;9	6 ;11 ;13	7 ;13 ;16
J_6	3 ;3 ;4	8 ;4 ;8	2 ;9 ;14	
J_7	3 ;4 ;6	8 ;8 ;13	4 ;13 ;16	
J_8	1 ;3 ;5	2 ;5 ;9	8 ;13 ;14	5 ;14 ;15

TAB. 2.10 – Valeurs des critères relatives au benchmark 8×8

C_1	C_2	C_3	C_4	C_5	C_μ
16	13	75	10.5	241	0.692

TAB. 2.11 – Valeurs des critères relatives au benchmark 10×10

C_1	C_2	C_3	C_4	C_5	C_μ
7	5	44	2.5	152	0.952

TAB. 2.12 – Valeurs des critères relatives au benchmark 10×7

C_1	C_2	C_3	C_4	C_5	C_μ
11	10	64	10	212	0.774

TAB. 2.13 – Valeurs des critères relatives au benchmark 15×10

C_1	C_2	C_3	C_4	C_5	C_μ
23	11	99	16,5	327	0.933

TAB. 2.14 – Solutions relatives au benchmark 10×10

	O_1	O_2	O_3		O_1	O_2	O_3		O_1	O_2	O_3
J_1	1 ;0 ;1	3 ;1 ;2	4 ;4 ;5	J_1	1 ;0 ;1	3 ;1 ;2	4 ;4 ;5	J_1	1 ;0 ;1	3 ;1 ;2	4 ;4 ;5
J_2	1 ;1 ;3	7 ;3 ;4	3 ;5 ;7	J_2	1 ;1 ;3	7 ;3 ;4	3 ;5 ;7	J_2	1 ;1 ;3	4 ;3 ;4	3 ;5 ;7
J_3	10 ;0 ;1	4 ;1 ;2	7 ;5 ;6	J_3	10 ;0 ;1	8 ;1 ;2	7 ;5 ;6	J_3	10 ;0 ;1	8 ;1 ;2	7 ;4 ;5
J_4	7 ;0 ;1	10 ;1 ;5	4 ;5 ;6	J_4	7 ;0 ;1	10 ;1 ;5	4 ;5 ;6	J_4	7 ;0 ;1	10 ;1 ;5	4 ;5 ;6
J_5	9 ;0 ;2	9 ;2 ;3	4 ;6 ;7	J_5	9 ;0 ;2	9 ;2 ;3	4 ;6 ;7	J_5	9 ;0 ;2	9 ;2 ;3	4 ;6 ;7
J_6	6 ;0 ;2	9 ;3 ;5	7 ;6 ;7	J_6	6 ;0 ;2	9 ;3 ;5	7 ;6 ;7	J_6	6 ;0 ;2	9 ;3 ;5	7 ;5 ;6
J_7	1 ;3 ;4	3 ;4 ;5	6 ;5 ;6	J_7	1 ;3 ;4	3 ;4 ;5	6 ;5 ;6	J_7	1 ;3 ;4	3 ;4 ;5	6 ;5 ;6
J_8	5 ;0 ;2	2 ;2 ;5	2 ;5 ;7	J_8	5 ;0 ;2	2 ;2 ;5	2 ;5 ;7	J_8	5 ;0 ;2	2 ;2 ;5	2 ;5 ;7
J_9	3 ;0 ;1	7 ;4 ;5	6 ;6 ;7	J_9	3 ;0 ;1	7 ;4 ;5	6 ;6 ;7	J_9	3 ;0 ;1	7 ;1 ;2	6 ;6 ;7
J_{10}	6 ;2 ;3	4 ;3 ;4	5 ;4 ;7	J_{10}	6 ;2 ;3	4 ;3 ;4	5 ;4 ;7	J_{10}	6 ;2 ;3	7 ;3 ;4	5 ;4 ;7

TAB. 2.15 – Solutions relatives au benchmark 10×7

	O_1	O_2	O_3		O_1	O_2	O_3
J_1	1 ;0 ;1	5 ;5 ;6	6 ;7 ;11	J_1	1 ;0 ;1	5 ;5 ;6	6 ;7 ;11
J_2	7 ;0 ;3	1 ;3 ;5		J_2	7 ;0 ;3	1 ;3 ;5	
J_3	7 ;3 ;4	3 ;4 ;5	5 ;6 ;8	J_3	7 ;3 ;4	3 ;4 ;5	5 ;6 ;8
J_4	1 ;1 ;3	6 ;3 ;6	5 ;8 ;9	J_4	1 ;1 ;3	6 ;3 ;6	1 ;7 ;8
J_5	5 ;0 ;1	1 ;5 ;7	2 ;7 ;8	J_5	5 ;0 ;1	1 ;5 ;7	2 ;7 ;8
J_6	3 ;0 ;4	3 ;5 ;6	3 ;6 ;8	J_6	3 ;0 ;4	3 ;5 ;6	3 ;6 ;8
J_7	7 ;4 ;6	7 ;6 ;9	3 ;9 ;11	J_7	7 ;4 ;6	7 ;6 ;9	3 ;9 ;11
J_8	2 ;0 ;1	4 ;1 ;9	2 ;9 ;11	J_8	2 ;0 ;1	4 ;1 ;9	2 ;9 ;11
J_9	5 ;1 ;5	2 ;5 ;7	4 ;9 ;11	J_9	5 ;1 ;5	2 ;5 ;7	4 ;9 ;11
J_{10}	2 ;1 ;5	6 ;6 ;7	1 ;7 ;8	J_{10}	2 ;1 ;5	6 ;6 ;7	1 ;8 ;9

TAB. 2.16 – Solutions relatives au benchmark 15×10

	O_1	O_2	O_3	O_4
J_1	1 ;5 ;6	2 ;13 ;14	3 ;14 ;15	6 ;17 ;21
J_2	7 ;3 ;4	3 ;4 ;6	10 ;15 ;16	8 ;18 ;19
J_3	7 ;6 ;7	7 ;15 ;16	2 ;16 ;18	4 ;18 ;19
J_4	5 ;0 ;1	5 ;10 ;11	6 ;16 ;17	10 ;20 ;21
J_5	5 ;9 ;10	6 ;10 ;12	7 ;16 ;18	6 ;21 ;23
J_6	2 ;7 ;8	1 ;14 ;15		
J_7	1 ;6 ;7	2 ;14 ;15		
J_8	7 ;7 ;8	8 ;15 ;16	1 ;17 ;20	1 ;20 ;21
J_9	10 ;8 ;9	3 ;9 ;11	4 ;11 ;16	8 ;19 ;23
J_{10}	10 ;9 ;10	1 ;15 ;17	10 ;17 ;18	10 ;21 ;23
J_{11}	7 ;14 ;15	6 ;15 ;16	3 ;16 ;18	9 ;21 ;23
J_{12}	8 ;13 ;15	5 ;15 ;19	9 ;19 ;21	7 ;21 ;23
J_{13}	2 ;11 ;13	10 ;13 ;15	10 ;18 ;20	2 ;20 ;22
J_{14}	1 ;12 ;14	9 ;14 ;16	8 ;16 ;18	5 ;19 ;21
J_{15}	9 ;5 ;10	3 ;11 ;13	3 ;18 ;20	5 ;21 ;23

	O_1	O_2	O_3	O_4
J_1	1 ;5 ;6	2 ;13 ;14	3 ;14 ;15	6 ;17 ;21
J_2	7 ;3 ;4	3 ;4 ;6	10 ;15 ;16	8 ;18 ;19
J_3	7 ;6 ;7	7 ;15 ;16	2 ;16 ;18	4 ;18 ;19
J_4	5 ;0 ;1	5 ;10 ;11	6 ;16 ;17	10 ;20 ;21
J_5	5 ;9 ;10	6 ;10 ;12	2 ;18 ;20	6 ;21 ;23
J_6	2 ;7 ;8	1 ;14 ;15		
J_7	1 ;6 ;7	2 ;14 ;15		
J_8	7 ;7 ;8	8 ;15 ;16	1 ;17 ;20	1 ;20 ;21
J_9	10 ;8 ;9	3 ;9 ;11	4 ;11 ;16	8 ;19 ;23
J_{10}	10 ;9 ;10	1 ;15 ;17	10 ;17 ;18	10 ;21 ;23
J_{11}	7 ;14 ;15	6 ;15 ;16	3 ;16 ;18	9 ;21 ;23
J_{12}	8 ;13 ;15	5 ;15 ;19	9 ;19 ;21	7 ;21 ;23
J_{13}	2 ;11 ;13	10 ;13 ;15	10 ;18 ;20	2 ;20 ;22
J_{14}	1 ;12 ;14	9 ;14 ;16	8 ;16 ;18	5 ;19 ;21
J_{15}	9 ;5 ;10	3 ;11 ;13	3 ;18 ;20	5 ;21 ;23

	O_1	O_2	O_3	O_4
J_1	1 ;5 ;6	2 ;13 ;14	3 ;14 ;15	6 ;17 ;21
J_2	7 ;3 ;4	3 ;4 ;6	10 ;15 ;16	8 ;18 ;19
J_3	7 ;6 ;7	7 ;15 ;16	2 ;16 ;18	4 ;18 ;19
J_4	5 ;0 ;1	5 ;10 ;11	6 ;16 ;17	10 ;20 ;21
J_5	5 ;9 ;10	6 ;10 ;12	2 ;18 ;20	6 ;21 ;23
J_6	2 ;7 ;8	1 ;14 ;15		
J_7	1 ;6 ;7	2 ;14 ;15		
J_8	7 ;7 ;8	8 ;15 ;16	1 ;17 ;20	1 ;20 ;21
J_9	10 ;8 ;9	3 ;9 ;11	4 ;11 ;16	8 ;19 ;23
J_{10}	10 ;9 ;10	1 ;15 ;17	10 ;17 ;18	10 ;21 ;23
J_{11}	7 ;14 ;15	6 ;15 ;16	3 ;16 ;18	9 ;21 ;23
J_{12}	8 ;13 ;15	5 ;15 ;19	9 ;19 ;21	7 ;21 ;23
J_{13}	2 ;11 ;13	10 ;13 ;15	10 ;18 ;20	2 ;20 ;22
J_{14}	1 ;12 ;14	9 ;14 ;16	8 ;16 ;18	5 ;19 ;21
J_{15}	9 ;5 ;10	3 ;11 ;13	3 ;18 ;20	5 ;21 ;23

	O_1	O_2	O_3	O_4
J_1	1 ;5 ;6	2 ;13 ;14	3 ;14 ;15	6 ;17 ;21
J_2	4 ;3 ;4	3 ;4 ;6	10 ;15 ;16	8 ;18 ;19
J_3	7 ;6 ;7	7 ;15 ;16	2 ;16 ;18	4 ;18 ;19
J_4	5 ;0 ;1	5 ;10 ;11	6 ;16 ;17	10 ;20 ;21
J_5	5 ;9 ;10	6 ;10 ;12	2 ;18 ;20	6 ;21 ;23
J_6	2 ;7 ;8	1 ;14 ;15		
J_7	1 ;6 ;7	2 ;14 ;15		
J_8	7 ;7 ;8	8 ;15 ;16	1 ;17 ;20	1 ;20 ;21
J_9	10 ;8 ;9	3 ;9 ;11	4 ;11 ;16	8 ;19 ;23
J_{10}	10 ;9 ;10	1 ;15 ;17	10 ;17 ;18	10 ;21 ;23
J_{11}	7 ;14 ;15	6 ;15 ;16	3 ;16 ;18	9 ;21 ;23
J_{12}	8 ;13 ;15	5 ;15 ;19	9 ;19 ;21	7 ;21 ;23
J_{13}	2 ;11 ;13	10 ;13 ;15	10 ;18 ;20	2 ;20 ;22
J_{14}	1 ;12 ;14	9 ;14 ;16	8 ;16 ;18	5 ;19 ;21
J_{15}	9 ;5 ;10	3 ;11 ;13	3 ;18 ;20	5 ;21 ;23

	O_1	O_2	O_3	O_4
J_1	1 ;5 ;6	2 ;13 ;14	3 ;14 ;15	6 ;17 ;21
J_2	4 ;3 ;4	3 ;4 ;6	10 ;15 ;16	8 ;18 ;19
J_3	7 ;6 ;7	7 ;15 ;16	2 ;16 ;18	4 ;18 ;19
J_4	5 ;0 ;1	5 ;10 ;11	6 ;16 ;17	10 ;20 ;21
J_5	5 ;9 ;10	6 ;10 ;12	2 ;18 ;20	6 ;21 ;23
J_6	2 ;7 ;8	1 ;14 ;15		
J_7	1 ;6 ;7	2 ;14 ;15		
J_8	7 ;7 ;8	8 ;15 ;16	1 ;17 ;20	1 ;20 ;21
J_9	10 ;8 ;9	3 ;9 ;11	4 ;11 ;16	8 ;19 ;23
J_{10}	10 ;9 ;10	1 ;15 ;17	10 ;17 ;18	10 ;21 ;23
J_{11}	7 ;14 ;15	6 ;15 ;16	3 ;16 ;18	9 ;21 ;23
J_{12}	8 ;13 ;15	5 ;15 ;19	9 ;19 ;21	7 ;21 ;23
J_{13}	2 ;11 ;13	10 ;13 ;15	10 ;18 ;20	2 ;20 ;22
J_{14}	1 ;12 ;14	9 ;14 ;16	8 ;16 ;18	5 ;19 ;21
J_{15}	9 ;5 ;10	3 ;11 ;13	3 ;18 ;20	5 ;21 ;23

2.7 Conclusion

Dans ce chapitre, nous avons mis en évidence notre contribution dans l'adaptation des algorithmes génétiques pour la résolution de nouvelle variantes de problèmes d'ordonnancement NP-difficiles, notamment, les problèmes d'ordonnancement job-shop flexible. Pour surmonter la complexité des problèmes posés et leur garantir des solutions admissibles, nous avons proposé une nouvelle approche d'évaluation multi-critères. Basée sur le principe de la logique floue et des bornes inférieures, cette approche vise l'homogénéisation des critères en vue d'harmoniser leurs valeurs numériques afin que l'un ne domine pas l'autre. En plus, l'adoption de cette approche permet d'offrir au décideur un ensemble de solutions réalisables. Quelques résultats des simulations élaborées sont présentés afin de montrer les performances de l'approche proposée. Cette approche bien qu'elle soit simple améliore des grands objectifs cruciaux qui consistent à :

- générer une variété de solutions optimales diversifiées dans l'espace de recherche de solutions ;
- aider le décideur quand il ne peut pas donner une préférence particulière à l'une des fonctions objectifs ;
- tenir compte du coût de la production.

Les recherches que nous avons menées sont théoriques, mais nous avons mis en évidence leur flexibilité par leur adaptation aisée aux changements qui touchent la configuration du système de fabrication et des conditions opératoires.

Dans le chapitre suivant, nous mettons l'accent sur les méthodes de transformation dynamiques des problèmes multi-objectifs en problèmes mono-objectif tout en minimisant les critères précédemment définis.

Chapitre 3

Optimisation multi-critères : Méthodes d'agrégation avec direction de recherche dynamique tenant compte du coût de la production

Sommaire

3.1 Introduction

Dans une économie libérale où l'arme est compétitivité, l'entreprise industrielle se trouve confrontée à des impératifs divers et souvent divergents : produire des quantités variables, au moindre coût dans les délais impartis et avec une qualité toujours meilleure.

Pour parvenir à trouver l'harmonie et l'équilibre nécessaires entre ces différents objectifs, les entreprises doivent disposer entre autre et surtout d'un système de production performant. Ces performances sont caractérisées par l'augmentation de la rentabilité, de la robustesse et de la souplesse de production, c'est-à-dire par la capacité d'adaptation qui rend un système de production flexible et réactif. L'intégration de tels critères doit être réalisée à deux niveaux différents et indissociables du système productif : d'une part les équipements qui permettent de concrétiser ou de réaliser matériellement la demande, d'autre part le système de pilotage qui prend en compte cette demande pour amener les ressources matérielles et humaines à la réaliser.

Dans ce contexte, si l'on veut répondre aux objectifs de coût, de qualité, de robustesse et d'évolution, la conception des systèmes de production doit être systématique, rigoureuse et donc être basée sur une méthodologie adaptée.

En effet, l'optimisation multi-objectifs cherche à optimiser plusieurs composantes d'un vecteur fonction coût. Contrairement à l'optimisation mono-objectif, la résolution d'un Problème Multi-Objectifs (PMO) n'a pas une solution unique, mais un ensemble de solutions, connu comme l'ensemble des solutions Paréto-Optimales (PO). Toute solution de cet ensemble est optimale dans le sens qu'aucune amélioration ne peut être faite sur une composante du vecteur sans dégradation d'au moins une autre composante du vecteur [Talbi, 1999]. Le choix d'une solution par rapport à une autre nécessite la connaissance du problème et de nombreux facteurs liés au problème. Ainsi, une solution choisie par un décideur peut ne pas être acceptable pour un autre. Il est donc utile d'avoir plusieurs alternatives dans le choix d'une solution Paréto-optimale (PO).

Dans ce chapitre, on s'intéresse aux problèmes d'ordonnancement dans les ate-

liers de production ou l'objectif principal est de produire le maximum de produits avec un temps et un coût minimums. On souhaite alors minimiser le Makespan et surtout le coût de production, en utilisant les algorithmes génétiques, ces dernier ayant été efficacement utilisés dans la résolution de problèmes multi-critères [Talbi, 1999], [Srinivas et Deb, 1995]. Pour la minimisation de ces critères, il existe plusieurs méthodes de transformation des problèmes multi-objectifs en problèmes mono-objectifs [Collette et Siarry, 2002]. Le but principal est de générer une variété de solutions Paréto-optimales diversifiées dans l'espace de recherche.

3.2 Definition du problème multi-critères

Un PMO peut-être défini de la manière suivante :

$$C(x) = (C_1(x), C_2(x), ..., C_{n_c}(x)) \quad \text{avec} \quad x \in S \tag{3.1}$$

où $n_c \geq 2$ est le nombre de fonctions objectifs, x est une solution réalisable et $C(x)$ est le vecteur des critères à optimiser, S représente l'ensemble des solutions réalisables associées à des contraintes d'égalités, d'inégalités et à des bornes explicites. Dans la résolution de PMO, plusieurs méthodes traditionnelles transforment le PMO en un problème mono-objectif. Parmi ces méthodes on trouve : la méthode d'agrégation, la méthode de compromis, et la méthode de programmation par but [Talbi, 1999], [Collette et Siarry, 2002]. Ces approches ont été largement utilisées dans la littérature à l'aide de différentes métaheuristiques telles que : algorithmes génétiques [Liu et al., 1998], recuit simulé [Serafini, 1992], recherche tabou [Glover et Laguna, 1997]...

Dans notre travail, les algorithmes génétiques sont utilisés pour le problème multi-objectifs dont le but de minimiser les cinq critères formulés dans le chapitre précédent par les équations (2.13) à (2.19). Dans ce cas, les méthodes d'agrégation avec recherche de direction dynamique sont utilisées pour aider le décideur quand il ne peut pas clairement donner une préférence particulière à un critère.

Principalement, nous pouvons distinguer deux classes : les approches Paréto-optimales et les approches non basées sur la notion de la Paréto dominance.

Les approches Paréto-optimales

La notion de Paréto dominance est l'une des approches les plus importantes utilisées dans l'optimisation multi-objectifs. L'optimalité au sens de Paréto a été proposée pour fournir de la flexibilité et afin de construire un grand ensemble de choix pour le décideur.

Les approches non basées sur la Paréto dominance

La littérature présente un grand ensemble de différentes approches. La première variante, appliquée dans beaucoup de travaux, est basée sur l'agrégation des différents critères dans un objectif simple (l'opérateur OWA). Selon le problème étudié, la littérature présente d'autres expressions et d'autres formulations pour agréger les différents objectifs. Néanmoins, dans la plupart des cas, il est difficile de concevoir une telle formulation, particulièrement quand les relations entre les objectifs sont en conflit et non linéaires. Ainsi, une étude rigoureuse de telles relations est nécessaire avant de décider la formulation.

Récemment, les algorithmes évolutionnistes ont été présentés comme une technique appropriée pour la résolution de problèmes d'optimisation multi-objectifs et basée sur les approches agrégatives. La littérature montre que de telles algorithmes ont été efficacement employés pour construire des approches performantes se basant sur la notion de la Paréto dominance.

3.3 Analyse floue multi-critères

Dans ce contexte décisionnel réel, les décideurs sont souvent amener à choisir une solution en prenant explicitement appui sur plusieurs critères généralement en conflit. Par ailleurs, chaque critère joue le rôle particulier dans la prise de décision. Néanmoins, connaître les préférences du décideur et déterminer les poids des critères sont bien souvent des tâches très délicates. La notation d'importance relative des

critères est centrale en aide multi-critères à la décision. Elle est utilisée dans une très grande majorité des méthodes proposées pouvant être réparties en deux principaux groupes [Mousseau, 1992].

– Les méthodes directes : ces méthodes affectent directement des poids aux critères en se basant sur une perception intuitive de l'importance relative des critères [Yager, 1981]. Elles présentent certains inconvénients que nous avons mis en évidence dans le chapitre précédent. En effet, pour le même ordre de préférence, des vecteurs poids différents peuvent donner des solutions différentes.
– Les méthodes indirectes : ces méthodes se basent sur l'évaluation des poids des critères par l'agrégation des informations fournies par le décideur. Ces informations doivent être fiables et exploitables [Pratyush et Yang, 1998]. Parmi ces méthodes, nous citons l'analyse hiérarchique [Saaty, 1980] qui est le plus répondue en ordonnancement. Elle est basée sur des comparaisons par paires de critères en ayant recours à un vocabulaire très limité. L'inconvénient de cette approche est que la quantification des jugements de l'expert peut mener à une incohérence entre les comparaisons de données par ce dernier et les valeurs numériques qui leurs sont associées.

3.4 Conception des algorithmes génétiques pour les problèmes d'ordonnancement

En général, les algorithmes génétiques sont des algorithmes d'optimisation qui reposent sur le hasard permettant de trouver une solution optimale. La première description du processus des algorithmes génétiques a été donnée par Holland en 1960 [Goldberg, 1994].

Les AGs sont des algorithmes itératifs de recherche dont le but est d'optimiser une fonction prédéfinie, appelée le critère ou fonction coût (fitness) ; ils travaillent en parallèle sur un ensemble de solutions candidates, appelé "population" d'individus ou chromosomes. Ces derniers sont constitués d'un ensemble d'éléments, appelés "gènes", qui peuvent prendre plusieurs valeurs appelées "allèles" [Renders, 1995]. Un

chromosome est une représentation ou un codage d'une solution du problème donné. Une première population est choisie soit aléatoirement, soit par des heuristiques ou par des méthodes spécifiques au problème, soit encore par mélange de solutions aléatoires et heuristiques ; cette population doit être suffisamment diversifiée pour que l'algorithme ne reste pas bloqué dans un optimum local. C'est ce qui se produit lorsque trop d'individus sont semblables. Les AGs gênèrent de nouveaux individus de telle sorte qu'ils soient plus performants que leurs prédécesseurs. Le processus d'amélioration des individus s'effectue par utilisation d'opérateurs génétiques qui sont : la selection, le croisement et la mutation [Goldberg, 1994].

Dans le cas général, un algorithme génétique a besoin de quatre composants fondamentaux [Yamada et Nakano, 1997] :

- une fonction de codage qui transforme les données de l'espace de recherche en données utilisables par un ordinateur : par exemple, une séquence de bits ou bien un nombre réel ;
- un moyen de créer une population initiale à partir des solutions potentielles ;
- une fonction qui permet d'évaluer l'adaptation d'un chromosome à son environnement ; ce qui offre la possibilité de comparer des individus. Cette fonction est en fait construite à partir du critère que l'on désire optimiser. L'application de cette fonction à un élément de la population donne sa *fitness* (fonction objectif) ;
- des opérateurs qui modifient les enfants après la reproduction.

3.4.1 Codage

Le codage par chromosome d'un ordonnancement consiste à modéliser la solution du problème par une structure de données informatique, représentant le plan de production, tout en tenant compte des contraintes de précédences, des contraintes temporelles et de ressources.

Plusieurs représentations par chromosomes de la solution d'ordonnancement ont été présentées dans la littérature [Mesghouni, 1999].

On distingue la représentation directe où le chromosome représente implicite-

ment la solution de l'ordonnancement [Ghedjati, 1994], la représentation indirecte indépendante du problème où le chromosome représente une solution de l'ordonnancement indépendamment des données du problème et la représentation indirecte spécifique au problème où le chromosome contient un ensemble d'informations spécifique à un problème donné tel que le temps d'exécution d'une opération, la date de début d'exécution, la machine correspondante,...[Mesghouni, 1999].

Pour la modélisation de l'ordonnancement en Job-Shop, le codage le plus approprié et riche du point de vue informationnelle, à notre sens est celui proposé par [Mesghouni, 1999], qui d'une part représente, clairement par une structure matricielle les contraintes de précédences, les contraintes temporelles sous forme date de début d'exécution de l'opération, l'affectation des machines aux opérations, et d'autre part présente une grande flexibilité pour les phases de sélection, reproduction et évaluation.

Un individu (ou chromosome) est représenté par une matrice, chaque ligne représentant la gamme de chaque ordre de fabrication. Chaque cellule de cette ligne (représentant une opération) contient trois termes. Le premier indique le numéro de la machine qui est affectée à l'exécution de cette opération, le second représente la date de début d'exécution de l'opération et le troisième représente la date de fin d'exécution de l'opération. Ces dates sont calculées en tenant compte des contraintes de ressource et de précédence. Ce codage proposé par [Mesghouni, 1999] est appelé Codage Parallèle d'Ordres Fabrication "CPOF", tableau 3.1 :

TAB. 3.1 – Codage Parallèle d'Ordres Fabrication "CPOF"

	O_1	O_2	O_i
P1	M_k ; $r_{1,1,k}$; $tf_{1,1,k}$	M_k ; $r_{2,1,k}$; $tf_{2,1,k}$	...
Pj	M_k ; $r_{1,j,k}$; $tf_{1,j,k}$	...	M_k ; $r_{i,j,k}$; $tf_{i,j,k}$

3.4.2 Opérateurs génétiques

3.4.2.1 Croisement

L'opérateur de croisement utilise deux individus (parents) pour créer un ou deux enfants. Cette étape consiste simplement à déterminer la structure génétique de l'enfant en fonction de celle des parents [Rebaudengo et Reorda, 1996]. Le but du croisement reste toujours à savoir obtenir par mélange de solutions d'autres chromosomes susceptibles d'améliorer le résultat. Dans notre cas, il y a deux opérateurs de croisement [Mesghouni, 1999] ; l'opérateur de croisement ligne manipule les jobs et l'opérateur de croisement colonne manipule un ensemble d'opérations.

3.4.2.2 Mutation

Le rôle essentiel de la mutation est d'introduire une certaine diversification dans la population que l'opérateur de croisement ne peut pas apporter. Nous envisageons de profiter de ce phénomène pour améliorer le résultat obtenu. Dans notre cas, un opérateur de mutation [Mesghouni, 1999] est utilisé ; a pour objectif d'équilibrer la charge des machines.

3.4.2.3 Sélection

La selection est l'un des éléments les plus importants dans l'algorithme génétique. Cet opérateur détermine quels individus dans la population qui vont transmettre ces gènes à la prochaine génération des individus. Dans notre cas, le principe de la roulette pondérée est utilisé, pour permettre de retenir les individus les plus prometteurs en terme de fonction coût.

3.5 Approches d'évaluation multi-critères

Nous présentons dans cette partie les approches évolutionnistes dynamiques tenant compte du coût de production que nous avons développée pour la gestion

de la production. En effet, nous nous proposons de traiter le problème d'optimisation multi-objectifs NP-difficile. Il est bien connu qu'une méthode d'optimisation multi-objectifs, pour qu'elle soit efficace, doit donner de bons résultats selon les préférences des décideurs [Kacem et al., 2001] ; elle doit donc résoudre le problème de l'évaluation des solutions et explorer intelligemment l'espace de recherche pour construire des solutions réalisables et satisfaisantes. Pour atteindre ces deux objectifs, nous proposons des approches hybrides qui tiennent compte de ces deux problèmes. Ainsi, beaucoup de considérations peuvent être prises pour tenir compte de toutes ces difficultés. Pour ce faire, deux approches sont proposées.

3.5.1 $1^{ière}$ approche : Approche d'évaluation multi-critères par Paréto-optimalité

Cette approche d'évaluation floue est basée sur les étapes suivantes.

– Pour chaque fonction objectif, une borne inférieure est calculée, telle que :

$$C_i(x) \geq C_i^b \ \ \forall x \in S, \quad 1 \leq i \leq n_c \tag{3.2}$$

où S est l'espace des solutions réalisables et n_c le nombre de fonctions objectif.

– Les valeurs des fonctions objectif, dans la plupart des cas, peuvent appartenir à différents intervalles de magnitude variable. En particulier, soit H une heuristique choisie et C_i^h la valeur maximale de la solution donnée par l'heuristique considérée selon la ième fonction objectif. La fuzzification est appliquée en utilisant les fonctions d'appartenance décrites sur la figure 3.1.

Un vecteur $C(x)$ est associé à chaque solution réalisable x, $C(x) \in \left[C_1^b, +\infty\right] \times ... \times \left[C_{n_c}^b, +\infty\right]$, avec $C(x) = (C_1(x), ..., C_{n_c}(x))^T$. Pour chaque vecteur $C(x)$, une fuzzification de ses composantes $C_i(x)$, proposée selon leurs positions dans les intervalles $\left[C_i^b,\ C_i^h\right]$, est considérée en deux sous-ensembles flous B^i et M^i, figure 3.1. On a :

$$\mu_i^B(C_i(x)) = \frac{C_i^h - C_i(x) + \varepsilon}{C_i^h - C_i^b + \varepsilon}, \text{ si } C_i(x) \in \left[C_i^b,\ C_i^h + \varepsilon\right] \tag{3.3}$$

$$\mu_i^B(C_i(x)) = 0, \text{ sinon}$$

$\mu_i^B(C_i(x))$ étant la mesure floue de $C_i(x)$ dans le sous-ensemble B^i.

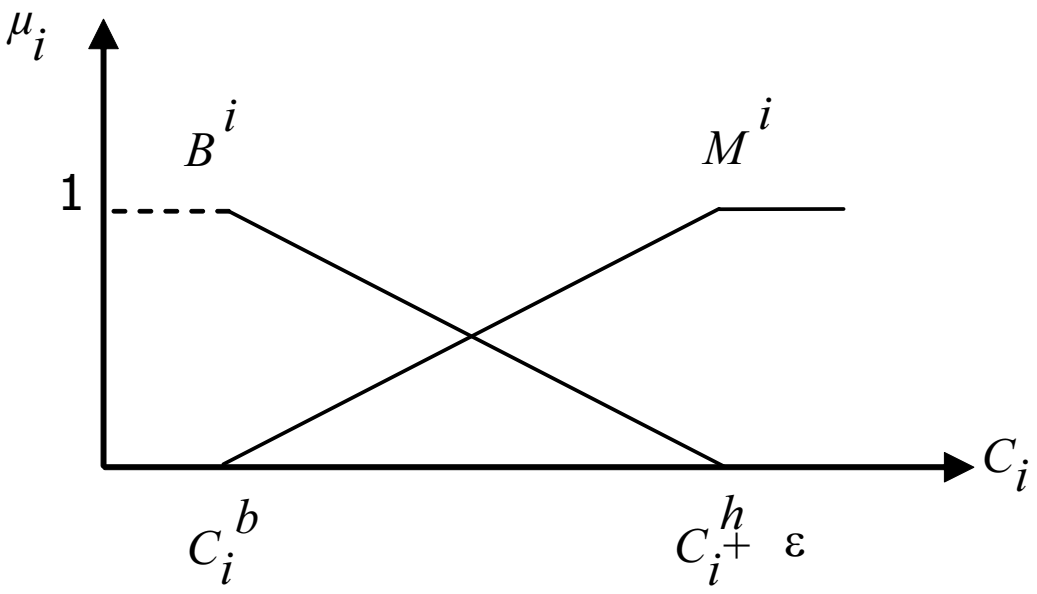

FIG. 3.1 – Application floue dans la résolution du problème d'échelle

Ensuite, la qualité de chaque solution x est caractérisée par le vecteur $C_B(x)$ défini dans 3.4 dont toutes les composantes sont homogènes puisqu'elles appartiennent toutes au même intervalle $[0, 1]$ et sont toutes sans dimension :

$$\begin{aligned} C_B(x) &= (a_1, ..., a_{n_c})^T \\ a_i &= \mu_i^B(C_i(x)), \quad \forall i = 1, 2, ..., n_c \end{aligned} \tag{3.4}$$

– Pour l'évaluation multi-critères, la fonction objectif $C_g(x)$ est réduite à la minimisation de la somme pondérée des critères, relative à l'utilisation de l'opérateur d'agrégation OWA [Yager, 1988] :

$$C_g(x) = \sum_{i=1}^{n_c} w_i a_i \tag{3.5}$$

Pour aider le décideur quand il ne peut pas clairement donner une préférence particulière à des fonctions objectif, un ensemble de solutions Paréto-optimales est construit, sans accorder de privilège à une direction particulière de recherche. Cette

approche est basée sur un algorithme dans lequel la fonction objectif $C_g(.)$, définie dans la relation (3.5), est utilisée pour l'évaluation des solutions. Les pondérations w_i $(1 \leq i \leq n_c)$ sont calculées en utilisant une règle floue. L'idée est de mesurer la qualité moyenne des solutions selon chaque critère à chaque itération et de calculer les différents poids suivant le degré de cette qualité. Le but est d'étudier les gains et les améliorations possibles des solutions en accordant la priorité à l'optimisation des fonctions objectif dont la moyenne des valeurs est loin de la borne inférieure ; cette approche est appelée approche agrégative avec direction de recherche dynamique [Saad et Benrejeb, 2006].

Soit $\bar{C}_i^k$ la moyenne des solutions de la i^{ieme} fonction objectif trouvée à la k^{ieme} itération :

$$\bar{C}_i^k = \frac{\sum_{x \in P_k} C_i^k(x)}{card(P_k)} \tag{3.6}$$

P_k étant la population des solutions à cette itération.

Pour chaque vecteur $C(x)$, une fuzzification est appliquée sur ses composantes $C_i(x)$ selon leurs positions dans les intervalles $\left[C_i^b,\ \bar{C}_i^0 + \varepsilon'\right]$ où ε' est une petite valeur positive introduite pour éviter le problème de la division par zéro ($\varepsilon' = 0.1C_i^b$, si $\bar{C}_i^0 = C_i^b$; $\varepsilon' = 0$ sinon).

L'évaluation de la qualité des solutions se fait en utilisant les fonctions d'appartenance définies dans la figure 3.2, relative aux deux sous-ensembles flous, *Proche* et *Loin* de la borne inférieure.

Les fonctions d'appartenance peuvent ainsi être formulées comme suit [Kacem, 2003] :

$$\mu_{ik}^L\left(\bar{C}_i^k\right) = \frac{\bar{C}_i^k - C_i^b}{\bar{C}_i^0 - C_i^b + \varepsilon'},\ \text{si}\ \bar{C}_i^k \in \left[C_i^b,\ \bar{C}_i^0 + \varepsilon'\right] \tag{3.7}$$

$$\mu_{ik}^L\left(\bar{C}_i^k\right) = 1,\ \text{sinon}$$

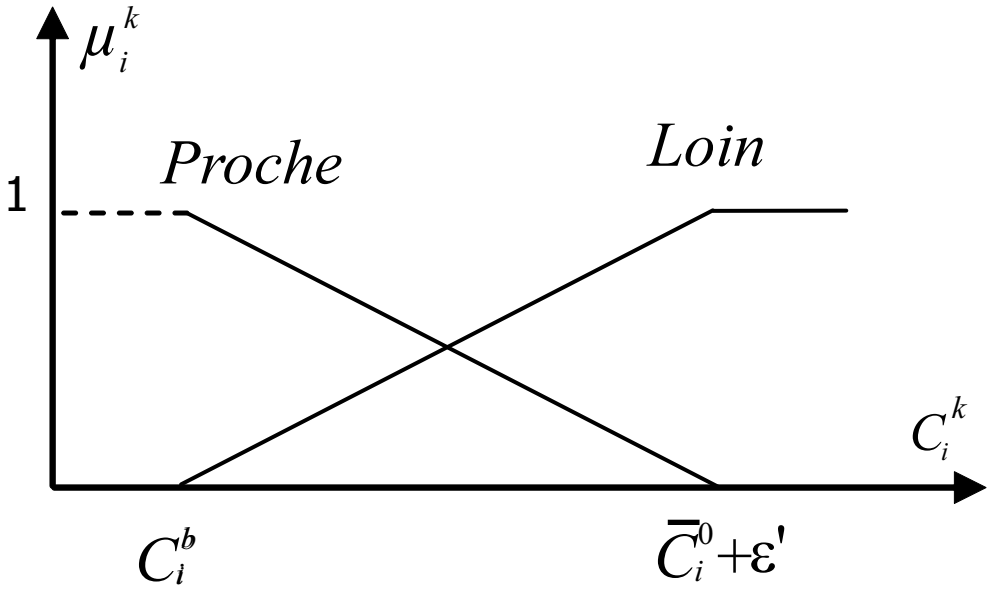

FIG. 3.2 – Fonction d'appartenance des différentes valeurs des critères

Le calcul des différentes pondérations w_i^{k+1} est effectué en utilisant les deux règles floues suivantes :

- Si (C_i^k est *Proche* de C_i^b) Alors (w_i^{k+1} diminue)
- Si (C_i^k est *Loin* de C_i^b) Alors (w_i^{k+1} augmente)

qui conduisent à l'expression de w_i^k suivante :

$$w_i^k = \frac{\mu_{ik}^L\left(\bar{C}_i^k\right)}{\sum_{j=1}^{n_c} \mu_{jk}^L\left(\bar{C}_j^k\right)},\ \forall i\ \forall k\ \text{ tq }\ 1 \leq i \leq n_c\ \text{ et }\ 2 \leq k \leq Q \tag{3.8}$$

w_i^1 correspond à la première itération définie par :

$$w_i^1 = \frac{1}{n_c},\ \forall i\ \text{ tq }\ 1 \leq i \leq n_c \tag{3.9}$$

Q désigne le nombre total d'itérations et L l'indice relatif au sous-ensemble flou *Loin*.

Les différents vecteurs de pondération $(W^1, W^2, ..., W^Q)$ sont calculés progressivement de la génération P_k à la génération P_{k+1}, selon la distance entre les bornes

inférieures et la moyenne des individus de la k^{ieme} génération, représentée par une cercle pleine noir dans la figure 3.3. Le but est d'améliorer les solutions en accordant la priorité à l'optimisation des fonctions objectif dont la moyenne des valeurs est loin de la borne inférieure. Il s'en suit qu'en utilisant une règle floue, il est envisageable de contrôler la direction de recherche afin de construire un ensemble final avec des solutions s'approchant le plus possible des valeurs optimales, figure 3.3.

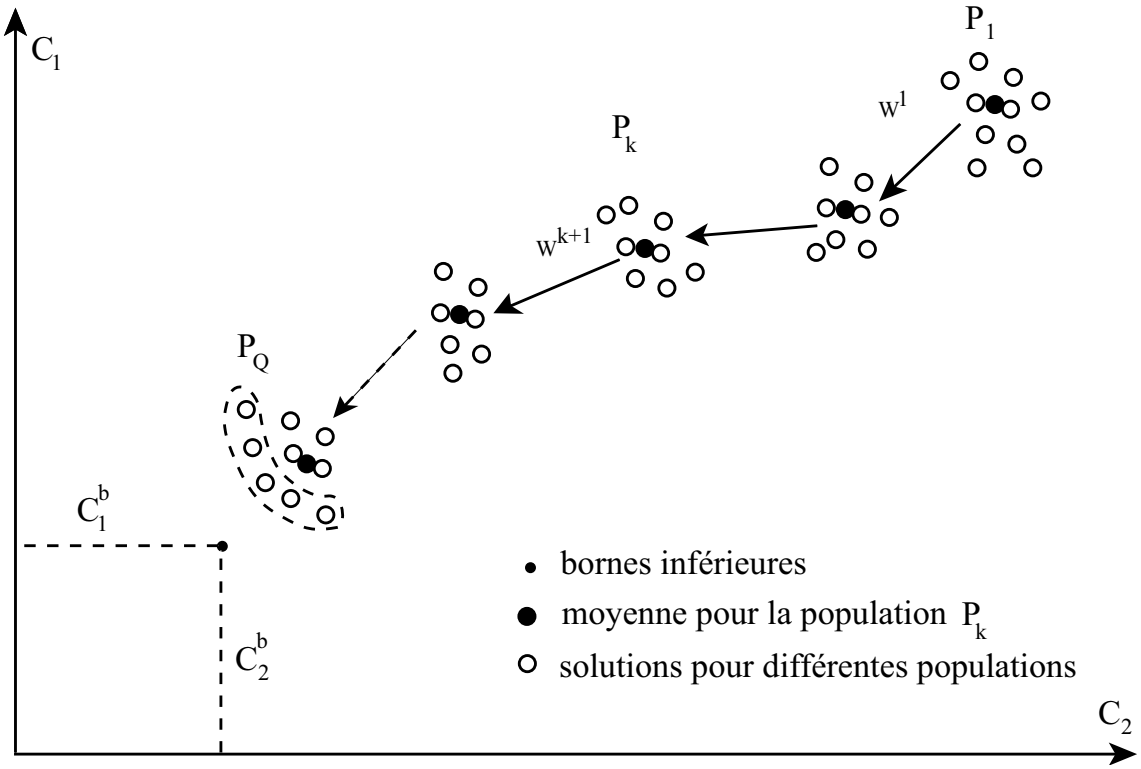

FIG. 3.3 – Direction de recherche

Cette méthode, généralement utilisée quand le décideur ne peut pas donner une préférence particulière à une fonction objectif, permet ainsi de générer des poids des critères différents d'une itération à une autre de manière dynamique en fonction de la valeur de la moyenne des solutions.

3.5.2 $2^{ième}$ approche : Evaluation globale par ε-dominance Paréto-optimalité

Comme pour l'ensemble des approches proposées, celle-ci traite séparément les deux problèmes de l'évaluation des solutions et de la recherche des nouvelles solutions dans des zones de l'espace de recherche non encore visitées. Notre préoccupation majeure lors de développement de cette méthode, est d'améliorer la première méthode développée auparavant, d'améliorer ainsi les solutions trouvées et d'apporter de l'aide aux experts de production (décideur). La souplesse et la simplicité de cette méthode permettent une gestion plus satisfaisante de la production des ateliers traités.

3.5.2.1 Détermination des bornes inférieures

Afin de réduire l'espace de recherche et de caractériser les limites des solutions réalisables, nous avons déterminé pour chaque critère une borne inférieure telle que :

$$C_i(x) \geq C_i^b \ \ \forall x \in S, \quad 1 \leq i \leq n_c \tag{3.10}$$

où S est l'espace des solutions réalisables et n_c le nombre de fonctions objectif.

Les valeurs de ces bornes inférieures correspondent à celles des critères en état normal. L'utilisation de ces bornes va nous permettre aussi d'évaluer et de juger les solutions obtenues avec précision.

3.5.2.2 Homogénéisation des objectifs

Pour pouvoir gommer l'influence de la difference entre les unités de mesure des différentes fonctions objectif, et aussi minimiser les effets dus aux différences de plages de variation de magnitude entre les fonctions objectif, nous utilisons une fonction écart f, définie de la manière suivante :

$$f(C_i(x)) = \frac{\mathrm{C}_i(\mathrm{x}) - C_i^b}{C_i^h - C_i^b} \tag{3.11}$$

Il vient :

$$f(C_i(x)) \in [0,\ 1] \quad \forall i = 1, ..., n_c$$

La qualité de chaque solution x est ensuite caractérisée par le vecteur $C_B(x)$ (3.12) dont toutes les composantes sont homogènes puisqu'elles appartiennent toutes au même intervalle $[0, 1]$ et sont toutes sans dimension :

$$\begin{aligned} &C_B(x) = (a_1,\ ...,\ a_{n_c})^T \\ &a_i = f(C_i(x)), \quad \forall i = 1,\ 2,\ ...,\ n_c \end{aligned} \tag{3.12}$$

3.5.2.3 Formulation de la fonction globale d'évaluation

La fonction d'évaluation globale utilisée $C_g(x)$, agrégeant de toutes les fonctions objectif, peut être formulée comme suit :

$$C_g(x) = \sum_{i=1}^{n_c} w_i a_i \tag{3.13}$$

où $w_i \in [0,\ 1], \quad \forall (1 \leq i \leq n_c)$ est le poids de la $i^{ième}$ fonction objectif tel que : $\sum_{i=1}^{n_c} w_i = 1$

Afin d'aider le décideur quand il ne peut pas clairement exprimer une préférence particulière à des fonctions objectif, un ensemble de solutions Paréto-optimales est construit sans accorder de privilège à une direction particulière de recherche. Cette approche est basée sur un algorithme dans lequel, la fonction objectif $C_g(.)$, définie dans la relation (3.13), est utilisée pour l'évaluation des solutions.

3.5.2.4 Détermination de la direction de recherche

Les pondérations $w_i \quad (1 \leq i \leq n_c)$ sont calculées en utilisant une règle floue. L'idée est de mesurer la qualité moyenne d'une solution selon chaque critère à chaque itération et de calculer les différents poids suivant le degré de cette qualité. Le but est d'étudier les gains et les améliorations possibles des solutions en accordant la

priorité à l'optimisation des fonctions objectif dont la moyenne des valeurs est loin de la borne inférieure.

Soit $\bar{C}_i^k$ la moyenne des solutions de la $i^{ième}$ fonction objectif trouvée, à la $k^{ième}$ itération :

$$f(\bar{C}_i^k) = \frac{\sum_{x \in P_k} f(C_i^k(x))}{card(P_k)} \tag{3.14}$$

P_k étant la population des solutions à cette itération.

Pour chaque vecteur $f(C(x))$, une fuzzification est appliquée sur ses composantes $f(C_i(x))$ selon leurs positions dans les intervalles $[0,\ \varepsilon]$ où ε est une petite valeur positive introduite pour évaluer la qualité ses solutions.

L'évaluation de la qualité des solutions se fait en utilisant les fonctions d'appartenance qui peuvent être formulées comme suit :

$$\begin{aligned} &\mu_{ik}^L\left(f(\bar{C}_i^k)\right) = \tfrac{f(\bar{C}_i^k)}{\varepsilon},\ \text{si}\ \ f(\bar{C}_i^k) \in [0,\ \varepsilon] \\ &\mu_{ik}^L\left(f(\bar{C}_i^k)\right) = 1,\ \text{sinon} \end{aligned} \tag{3.15}$$

Le calcul des différentes pondérations w_i^{k+1} est effectué en utilisant les deux règles floues suivantes :

- Si ($|f(C_i) - \varepsilon| > 0$) alors (w_i^{k+1} augmente)
- Si ($|f(C_i) - \varepsilon| = 0$) alors (w_i^{k+1} diminue)

qui conduisent à l'expression de w_i^k suivante :

$$w_i^k = \frac{\mu_{ik}^L\left(f(\overline{C}_i^k)\right)}{\sum_{j=1}^{n_c} \mu_{jk}^L\left(f(\overline{C}_j^k)\right)},\ \forall i\ \forall k\ \ \text{tq}\ \ 1 \leq i \leq n_c\ \ \text{et}\ \ 2 \leq k \leq Q \tag{3.16}$$

w_i^1 correspond à la première itération définie par :

$$w_i^1 = \frac{1}{n_c},\ \forall i\ \ 1 \leq i \leq n_c \tag{3.17}$$

avec Q le nombre total d'itérations et L l'indice relatif au sous-ensemble flou *Loin*.

Les différents vecteurs de pondération $(W^1, W^2, ..., W^Q)$ sont dynamiquement calculés (en évoluant d'une génération à une autre), selon la distance entre les bornes inférieures et la moyenne des individus la k^{ieme} génération, de la génération P_k à la génération P_{k+1}, représentée par une cercle pleine noir dans la figure 3.3. Le but est d'étudier les gains et les améliorations possibles des solutions en accordant la priorité à l'optimisation des fonctions objectifs dont la moyenne des valeurs est loin de la valeur optimale (ou de la borne inférieure). Il s'en suit qu'en utilisant une règle floue, il est envisageable de contrôler la direction de recherche afin de construire un ensemble final avec des solutions s'approchant le plus possible des valeurs optimales, figure 3.3. Cet ensemble final est utilisé pour déterminer les solutions dominantes au sens de Paréto ; les autres solutions sont rejetées.

Cette méthode, généralement utilisée quand le décideur ne peut pas donner une préférence particulière à une fonction objectif, permet ainsi de générer des poids des critères différents d'une itération à une autre de manière dynamique en fonction de la moyenne des solutions et des écarts des critères par rapport à ε (ε-dominance).

Cette méthode exploite les propriétés de la méthode d'optimisation par Paréto optimalité pour la détermination des poids des critères d'une manière dynamique et la technique ε-dominance pour mesurer la qualité des solutions par rapport aux des fonctions objectif considérées.

3.6 Mise en œuvre par simulation

Dans ce paragraphe, nous présentons quelques résultats de simulation des approches proposées dans la section précédente. Plusieurs problèmes pratiques ont été traités pour étudier la qualité de ces approches. Pour chaque benchmark, nous présentons les valeurs des critères pour chaque solution trouvée ainsi que les valeurs des bornes inférieures correspondantes.

Remarque 1 *Pour des raisons de simplicité de représentations, les différentes solutions sont représentées ici sous forme de codage "CPOF".*

3.6.1 Simulation de la première approche d'évaluation par Paréto-optimalité

Quatre benchmarks de type Job-Shop flexible $N \times M$, traitant N produits sur M machines [Xia et Wu, 2005], sont traités ici pour montrer la validité de la mise en œuvre de la méthode d'optimisation multi-critères proposée dans le paragraphe précédent Les coûts de pénalité et le coût des matières premières sont, dans les deux cas, les suivants (en unité monétaire) :

$$Mp_j = 2,\ h_j = 0,5\ et\ b_j = 2,\ j = 1, 2, ..., N$$
$$Cm_k = 3,\ k = 1, 2, ..., M$$

Les critères à optimiser au nombre de 5 sont formulés par les relations (2.13) à (2.16) et (2.19).

TAB. 3.2 – Benchmarks étudier par la première approche d'évaluation par Paréto-optimalité

$N \times M$	durées opératoires	solutions
8×8	table A.3	table 3.7
10×10	table A.4	table 3.6
10×7	table A.5	table 3.5
15×10	table A.6	table 3.8

Les différentes valeurs des critères données par la méthode d'optimisation multi-critères par Paréto-optimlité montrent son efficacité ainsi qu'il apparaît sur la table 3.3. Les valeurs des critères pour la frontière sont assez proches de celles des bornes inférieures. En effet, une telle approche permet de générer des solutions Paréto-optimales de bonne qualité.

Les différents résultats obtenus par l'utilisation de la méthode d'optimisation multi-critères par Paréto-optimlité sont présentés et comparés avec d'autres méthodes dans la table 3.4 : la colonne notée 'AL' est relative à l'approche par locali-

TAB. 3.3 – Résumé des résultats obtenus pour les benchmarks étudiés par la première approche proposée

$N \times M$	C_1^b	C_2^b	C_3^b	C_4^b	C_5^b	C_1	C_2	C_3	C_4	C_5	C_g
8×8	12	10	73	0	229	16	12	77	11.5	247	0,591
10×10	7	5	41	0	139	7	6	44	4	152	0,853
10×7	11	9	60	0	188	12	11	63	12,5	209	0,673
15×10	23	10	91	0	287	24	11	94	17	312	0,933

TAB. 3.4 – Comparaison des résultats obtenus pour les benchmarks étudiés par la première approche proposée

	AL			PSO+SA			Méthode proposée		
$N \times M$	C_1	C_2	C_3	C_1	C_2	C_3	C_1	C_2	C_3
8×8	16	13	75	16	13	73	16	12	77
10×10	7	5	45	7	6	44	7	6	44
10×7	15	11	61				12	11	63
15×10	23	11	95				24	11	94

sation [Kacem et al., 2002] et la colonne suivante 'PSO+SA' est relative à l'optimisation par recuit simulé [Xia et Wu, 2005]. Les différents résultats montrent que les solutions obtenues sont généralement acceptables et satisfaisantes, malgré que cinq critères soient considérés au lieu de trois dans [Kacem et al., 2002] et [Xia et Wu, 2005]. Les valeurs des fonctions objectifs montrent l'efficacité de l'approche proposée, table 3.3.

L'optimisation multi-critères par Paréto-optimlité s'effectue en deux parties : une partie d'évaluation et une partie de résolution. La première, vise à élaborer un moyen de mesure de la qualité des solutions et à intégrer par utilisation de la description floue des préférences subjectives dans un cadre coopératif. Elle peut conduire à des

solutions dominantes, au sens Paréto, en exploitant l'ensemble de bornes inférieures proposé. Dans la seconde partie, représentant le noyau de l'approche pour l'optimisation multi-critères des ateliers flexibles de type job-shop, il s'est avéré possible d'intégrer au mieux les difficultés du problème et de tenir compte des préférences du décideur qui ne sont pas forcément homogènes. Elle permet aussi de générer des solutions admissibles tout en étudiant les relations de dominance possibles entre les solutions candidates et de déterminer une ou un ensemble de solutions dominantes selon les critères considérés ou selon une agrégation de tels critères.

TAB. 3.5 – Solutions relatives au benchmark 10×7

	O_1	O_2	O_3
J_1	1 ;0 ;1	6 ;1 ;2	6 ;6 ;10
J_2	7 ;0 ;3	1 ;3 ;5	
J_3	7 ;3 ;4	6 ;4 ;5	5 ;10 ;12
J_4	1 ;1 ;3	4 ;3 ;4	1 ;7 ;8
J_5	2 ;0 ;1	1 ;5 ;7	2 ;9 ;10
J_6	7 ;4 ;8	3 ;8 ;9	3 ;9 ;11
J_7	3 ;0 ;3	7 ;8 ;11	6 ;11 ;12
J_8	4 ;0 ;1	5 ;1 ;10	2 ;10 ;12
J_9	3 ;3 ;7	2 ;7 ;9	4 ;9 ;11
J_{10}	2 ;1 ;5	6 ;5 ;6	1 ;8 ;9

3.6.2 Comparaison de la première approche avec celle basée sur l'utilisation de l'intégrale de Choquet

Cette approche a été comparée avec la méthode d'optimisation multi-critères à base de l'intégrale de Choquet [Saad et al., 2007a] développée au deuxième Chapitre. Cinq problèmes (4×5, 8×8, 10×10, 10×7 et 15×10) ont été traités pour étudier la qualité de ces deux approches. Le tableau 3.10 résume les résultats obtenus.

Dans ce paragraphe, nous montrons à l'aide des problèmes pratiques la robustesse de l'approche agrégative proposée. Nous concluons que les valeurs des critères pour les

TAB. 3.6 – Solutions relatives au benchmark 10×10

	O_1	O_2	O_3
J_1	1 ;0 ;1	2 ;1 ;2	4 ;3 ;4
J_2	1 ;1 ;3	7 ;3 ;4	3 ;5 ;7
J_3	10 ;0 ;1	4 ;1 ;2	7 ;5 ;6
J_4	9 ;0 ;1	3 ;1 ;4	10 ;4 ;6
J_5	9 ;1 ;3	9 ;3 ;4	4 ;4 ;5
J_6	6 ;0 ;2	9 ;4 ;6	7 ;6 ;7
J_7	1 ;3 ;4	3 ;4 ;5	6 ;5 ;6
J_8	5 ;0 ;2	2 ;2 ;5	2 ;5 ;7
J_9	6 ;2 ;3	7 ;4 ;5	6 ;6 ;7
J_{10}	7 ;0 ;2	4 ;2 ;3	4 ;5 ;7

solutions suivent les poids y attribués et correspondant aux préférences du décideur. Les différentes valeurs des solutions données par notre approche sont résumées dans le tableau 3.10 ainsi que les valeurs des bornes inférieures. L'obtention de la solution optimale est donc expliquée par la qualité de l'approche Paréto-optimalité. Cette méthode de résolution permet de se rapprocher des valeurs des bornes inférieures tout en suivant les coefficients de pondération choisis.

Les solutions trouvées sont généralement d'une bonne qualité puisque l'écart entre les valeurs des critères et les valeurs des bornes inférieures est souvent faible ; il serait toutefois intéressant d'approfondir la recherche pour étudier les éventuelles relations entre un tel écart et la dispersion des valeurs des variables pour un problème donné.

3.6.3 Application de la première approche dans l'industrie agro-alimentaires

Cette méthode de résolution du problème d'ordonnancement dynamique a été appliquée dans l'industrie agro-alimentaire pour un atelier flexible de production à une machine. La méthode de résolution, consiste d'abord à explorer les différents cas d'ordonnancements réalisables d'un ensemble d'opérations candidates à être or-

TAB. 3.7 – Solutions relatives au benchmark 8×8

	O_1	O_2	O_3	O_4
J_1	4; 0; 3	5; 3; 6	6; 6; 8	
J_2	3; 0; 3	4; 3; 5	7; 9; 10	5; 10; 14
J_3	7; 0; 2	4; 5; 9	1; 9; 10	
J_4	2; 0; 1	6; 1; 6	1; 10; 14	
J_5	1; 0; 3	7; 3; 9	6; 9; 11	7; 11; 14
J_6	3; 3; 4	8; 4; 8	2; 9; 14	
J_7	3; 4; 6	8; 8; 13	4; 13; 16	
J_8	1; 3; 5	2; 5; 9	8; 13; 14	5; 14; 15

	O_1	O_2	O_3	O_4
J_1	5; 0; 3	5; 3; 6	6; 6; 8	
J_2	3; 0; 3	4; 3; 5	7; 9; 10	5; 10; 14
J_3	7; 0; 2	4; 5; 9	1; 9; 10	
J_4	2; 0; 1	6; 1; 6	1; 10; 14	
J_5	1; 0; 3	7; 3; 9	6; 9; 11	7; 11; 14
J_6	3; 3; 4	8; 4; 8	2; 9; 14	
J_7	3; 4; 6	8; 8; 13	4; 13; 16	
J_8	1; 3; 5	2; 5; 9	8; 13; 14	5; 14; 15

donnancées et d'identifier le meilleur, minimisant le coût total à savoir le coût des produits périmés et le coût de discount de distribution, deux critères spécifiques à l'industrie agro-alimentaire [Tangour et Saad, 2006].

Les résultats obtenus montrent que les solutions calculées sont généralement, acceptables et satisfaisantes. Les valeurs des différentes fonctions objectif montrent l'efficacité de l'approche proposée. De plus, la méthode proposée, a permis d'obtenir de bons résultats dans un temps de calcul polynomial. En fait, les diverses valeurs des critères indiqués par la méthode d'optimisation multi-objectifs par Paréto-optimalité montrent son efficacité. Les valeurs des critères pour la frontière de Paréto sont à proximité des bornes inférieures. En effet, une telle approche permet de trouver des solutions réalisables Paréto-optimales de bonne qualité.

Les exemples traités ont montré l'importance du coût de discount de distribution relativement au coût des produits périmés dans la prise de décision finale concernant le choix de l'ordonnancement dont la fonction coût est minimale. Ainsi, cette décision peut éviter la péremption de certains composants et produits semi-finis.

TAB. 3.8 – Solution relative au benchmark 15×10

	O_1	O_2	O_3	O_4
J_1	1 ;5 ;6	1 ;16 ;17	3 ;18 ;19	6 ;19 ;23
J_2	4 ;3 ;4	3 ;4 ;6	10 ;15 ;16	4 ;18 ;19
J_3	2 ;6 ;7	7 ;15 ;16	2 ;16 ;18	4 ;19 ;20
J_4	5 ;0 ;1	5 ;10 ;11	6 ;16 ;17	10 ;21 ;22
J_5	5 ;9 ;10	6 ;10 ;12	2 ;18 ;20	4 ;20 ;22
J_6	2 ;7 ;8	1 ;17 ;18		
J_7	1 ;6 ;7	1 ;18 ;20		
J_8	7 ;2 ;3	8 ;15 ;16	4 ;16 ;18	1 ;20 ;21
J_9	10 ;8 ;9	3 ;9 ;11	7 ;16 ;20	8 ;20 ;24
J_{10}	10 ;9 ;10	8 ;16 ;18	10 ;18 ;19	10 ;22 ;24
J_{11}	7 ;14 ;15	6 ;15 ;16	9 ;16 ;18	7 ;20 ;21
J_{12}	8 ;13 ;15	5 ;15 ;19	9 ;19 ;21	7 ;21 ;23
J_{13}	2 ;11 ;13	10 ;13 ;15	10 ;19 ;21	2 ;21 ;23
J_{14}	1 ;12 ;14	9 ;14 ;16	8 ;18 ;20	5 ;20 ;22
J_{15}	1 ;14 ;16	3 ;16 ;18	3 ;19 ;21	5 ;22 ;24

3.6.4 Simulation de la deuxième approche d'évaluation par ε-dominance Paréto-optimalité

Six benchmarks de type job-shop flexible $N \times M$, traitant N produits sur M machines, sont présentés ici pour montrer la validité de la mise en œuvre de la méthode d'optimisation multi-critères par Paréto ε-dominance. Les coûts des matières premières sont, dans les deux cas, les suivants (en unité monétaire) :

$$Mp_j = 2,\ j = 1, 2, ..., N$$
$$Cm_k = 3,\ k = 1, 2, ..., M$$

Les critères à optimiser au nombre de 4 sont formulés par les relations (2.13) à (2.15) et (2.19).

TAB. 3.9 – Comparaison des résultats obtenus par les deux méthodes proposées

$N \times M$	AL			Intégrale de Choquet					Paréto-Optimalté				
	C_1	C_2	C_3	C_1	C_2	C_3	C_4	C_5	C_1	C_2	C_3	C_4	C_5
4×5	16	9	35	16	8	32	0	104	16	8	32	0	104
8×8	16	13	75	16	13	75	10.5	241	16	14	75	11.5	247
10×10	7	5	45	7	5	44	2.5	152	7	6	44	4	152
10×7	15	11	61	16	11	68	16.5	224	17	12	64	6.5	212
15×10	23	11	95	23	11	100	17.5	330	24	11	94	17	312

TAB. 3.10 – Benchmarks étudier par la deuxième approche d'évaluation par ε-dominance Paréto-optimalité

$N \times M$	durées opératoires	solutions
3×6	table A.2	table 3.13
4×5	table A.1	table 3.14
8×8	table A.3	table 3.15
10×10	table A.4	table 3.17
15×10	table A.6	table 3.16

Les différentes valeurs des critères données par la méthode d'optimisation multi-critères par ε-dominance Paréto-optimalité montrent son efficacité ainsi qu'il apparaît sur la table 3.11. Les valeurs des critères pour la frontière sont assez proches de celles des bornes inférieures. En effet, une telle approche permet de générer des solutions Paréto-optimales de très bonne qualité.

Les différents résultats obtenus par l'utilisation de la méthode d'optimisation multi-critères par ε-dominance Paréto-optimlité sont présentés et comparés avec d'autres méthodes dans la table 3.12 : la colonne notée 'AL' est relative à l'approche par localisation [Kacem et al., 2002]. Les différents résultats montrent que les solutions obtenues sont généralement acceptables et satisfaisantes, malgré que quatre critères

TAB. 3.11 – Résumé des résultats obtenus pour les benchmarks étudiés par la deuxième approche proposée

$N \times M$	C_1^b	C_2^b	C_3^b	C_4^b	C_1	C_2	C_3	C_4
15×10	23	10	91	287	24	11	91	303
10×10	7	5	41	139	8	7	41	143
10×7	11	9	60	188	14	12	60	200
8×8	12	10	73	229	16	14	75	241
4×5	16	7	32	84	16	8	32	104
3×6	18	8	45	130	18	12	45	141

TAB. 3.12 – Comparaison des résultats obtenus pour les benchmarks étudiés la deuxième approche proposée

	AL			Méthode proposée		
$N \times M$	C_1	C_2	C_3	C_1	C_2	C_3
15×10	23	11	95	24	11	91
10×10	7	5	45	8	7	41
8×8	16	13	75	16	14	75
4×5	16	9	35	16	8	32

soient considérés au lieu de trois dans [Kacem et al., 2002]. Les valeurs des fonctions objectifs montrent l'efficacité de l'approche proposée, table 3.11.

TAB. 3.13 – Solutions relatives au benchmark 3×6

	O_1	O_2	O_3	O_4
J_1	6 ;0 ;1	1 ;1 ;5	2 ;5 ;10	6 ;10 ;13
J_2	4 ;0 ;6	5 ;6 ;10	5 ;10 ;18	
J_3	3 ;0 ;5	6 ;5 ;8	1 ;8 ;14	

TAB. 3.14 – Solutions relatives au benchmark 4×5

	O_1	O_2	O_3	O_4
J_1	4 ;1 ;2	2 ;2 ;6	4 ;6 ;10	
J_2	1 ;5 ;7	5 ;7 ;12	1 ;12 ;16	
J_3	3 ;1 ;7	2 ;7 ;8	4 ;10 ;12	4 ;12 ;13
J_4	1 ;7 ;8	2 ;8 ;9		

TAB. 3.15 – Solutions relatives au benchmark 8×8

	O_1	O_2	O_3	O_4
J_1	5 ;0 ;3	5 ;3 ;6	6 ;6 ;8	
J_2	3 ;0 ;3	4 ;3 ;5	7 ;5 ;6	5 ;6 ;10
J_3	7 ;0 ;2	3 ;6 ;12	1 ;12 ;13	
J_4	2 ;0 ;1	6 ;1 ;6	3 ;12 ;14	
J_5	1 ;0 ;3	4 ;5 ;9	6 ;9 ;11	7 ;11 ;14
J_6	3 ;3 ;4	8 ;4 ;8	2 ;9 ;14	
J_7	3 ;4 ;6	8 ;8 ;13	4 ;13 ;16	
J_8	1 ;3 ;5	2 ;5 ;9	8 ;13 ;14	5 ;14 ;15

TAB. 3.16 – Solutions relatives au benchmark 15×10

	O_1	O_2	O_3	O_4
J_1	1 ;5 ;6	1 ;16 ;17	3 ;17 ;18	9 ;18 ;22
J_2	4 ;3 ;4	3 ;4 ;6	10 ;15 ;16	8 ;20 ;21
J_3	2 ;6 ;7	9 ;7 ;8	2 ;14 ;16	4 ;18 ;19
J_4	5 ;0 ;1	5 ;10 ;11	6 ;18 ;19	10 ;21 ;22
J_5	5 ;9 ;10	6 ;10 ;12	7 ;15 ;17	4 ;19 ;21
J_6	2 ;7 ;8	1 ;17 ;18		
J_7	1 ;6 ;7	2 ;13 ;14		
J_8	7 ;2 ;3	8 ;15 ;16	4 ;16 ;18	1 ;18 ;19
J_9	10 ;8 ;9	3 ;9 ;11	7 ;17 ;21	10 ;22 ;24
J_{10}	10 ;9 ;10	8 ;16 ;18	10 ;18 ;19	2 ;19 ;21
J_{11}	7 ;14 ;15	6 ;15 ;16	9 ;16 ;18	7 ;21 ;22
J_{12}	8 ;13 ;15	5 ;15 ;19	6 ;19 ;21	7 ;22 ;24
J_{13}	2 ;11 ;13	10 ;13 ;15	10 ;19 ;21	2 ;21 ;23
J_{14}	1 ;12 ;14	9 ;14 ;16	8 ;18 ;20	5 ;20 ;22
J_{15}	1 ;14 ;16	6 ;16 ;18	3 ;18 ;20	5 ;22 ;24

TAB. 3.17 – Solutions relatives au benchmark 10×10

a	O_1	O_2	O_3
J_1	1 ;0 ;1	3 ;1 ;2	4 ;2 ;3
J_2	1 ;1 ;3	10 ;3 ;4	3 ;5 ;7
J_3	10 ;0 ;1	8 ;1 ;2	7 ;4 ;5
J_4	7 ;0 ;1	2 ;1 ;3	4 ;3 ;4
J_5	9 ;0 ;2	9 ;2 ;3	4 ;4 ;5
J_6	6 ;0 ;2	9 ;3 ;5	9 ;5 ;6
J_7	1 ;3 ;4	3 ;4 ;5	6 ;5 ;6
J_8	5 ;0 ;2	2 ;3 ;6	2 ;6 ;8
J_9	3 ;0 ;1	7 ;1 ;2	6 ;6 ;7
J_{10}	6 ;2 ;3	7 ;3 ;4	4 ;5 ;7

b	O_1	O_2	O_3
J_1	1 ;0 ;1	3 ;1 ;2	4 ;2 ;3
J_2	1 ;1 ;3	10 ;3 ;4	3 ;5 ;7
J_3	10 ;0 ;1	4 ;1 ;2	7 ;4 ;5
J_4	7 ;0 ;1	2 ;1 ;3	4 ;3 ;4
J_5	9 ;0 ;2	9 ;2 ;3	4 ;4 ;5
J_6	6 ;0 ;2	9 ;3 ;5	9 ;5 ;6
J_7	1 ;3 ;4	3 ;4 ;5	6 ;5 ;6
J_8	5 ;0 ;2	2 ;3 ;6	2 ;6 ;8
J_9	3 ;0 ;1	7 ;1 ;2	6 ;6 ;7
J_{10}	6 ;2 ;3	7 ;3 ;4	4 ;5 ;7

c	O_1	O_2	O_3
J_1	1 ;0 ;1	3 ;1 ;2	4 ;2 ;3
J_2	1 ;1 ;3	10 ;3 ;4	3 ;5 ;7
J_3	10 ;0 ;1	4 ;1 ;2	7 ;4 ;5
J_4	7 ;0 ;1	2 ;1 ;3	4 ;3 ;4
J_5	9 ;0 ;2	9 ;2 ;3	4 ;4 ;5
J_6	6 ;0 ;2	9 ;3 ;5	7 ;5 ;6
J_7	1 ;3 ;4	3 ;4 ;5	6 ;5 ;6
J_8	5 ;0 ;2	2 ;3 ;6	2 ;6 ;8
J_9	3 ;0 ;1	7 ;1 ;2	6 ;6 ;7
J_{10}	6 ;2 ;3	7 ;3 ;4	4 ;5 ;7

d	O_1	O_2	O_3
J_1	1 ;0 ;1	3 ;1 ;2	4 ;2 ;3
J_2	1 ;1 ;3	10 ;3 ;4	3 ;5 ;7
J_3	10 ;0 ;1	8 ;1 ;2	7 ;4 ;5
J_4	7 ;0 ;1	2 ;1 ;3	4 ;3 ;4
J_5	9 ;0 ;2	9 ;2 ;3	4 ;4 ;5
J_6	6 ;0 ;2	9 ;3 ;5	9 ;5 ;6
J_7	1 ;3 ;4	3 ;4 ;5	4 ;5 ;6
J_8	5 ;0 ;2	2 ;3 ;6	2 ;6 ;8
J_9	3 ;0 ;1	7 ;1 ;2	6 ;3 ;4
J_{10}	6 ;2 ;3	7 ;3 ;4	4 ;6 ;8

e	O_1	O_2	O_3
J_1	1 ;0 ;1	3 ;1 ;2	4 ;2 ;3
J_2	1 ;1 ;3	10 ;3 ;4	3 ;5 ;7
J_3	10 ;0 ;1	4 ;1 ;2	7 ;4 ;5
J_4	7 ;0 ;1	2 ;1 ;3	4 ;3 ;4
J_5	9 ;0 ;2	9 ;2 ;3	4 ;4 ;5
J_6	6 ;0 ;2	9 ;3 ;5	9 ;5 ;6
J_7	1 ;3 ;4	3 ;4 ;5	4 ;5 ;6
J_8	5 ;0 ;2	2 ;3 ;6	2 ;6 ;8
J_9	3 ;0 ;1	7 ;1 ;2	6 ;3 ;4
J_{10}	6 ;2 ;3	7 ;3 ;4	4 ;6 ;8

3.7 Conclusion

Dans ce chapitre, nous nous sommes intéressés aux différentes démarches de résolution du problème du job-shop flexible dans le cas où les machines sont supposées toujours disponibles. Cinq critères on été considérées : le Makespan, la charge critique (alléger la charge de toutes les machines), la charge totale, la pénalité de retard/avance (pénalités de retard et de stockage) et le coût de production, critère peu étudié pour ce type de problème.

Notre apport se situe essentiellement dans la résolution du problème d'affectation et de décision. En effet, selon le critère étudié, nous avons défini des critères spécifiques et appropriés que nous avons cherché à optimiser. Nous avons, par ailleurs, trouvé une estimation du critère global pour une solution d'affectation donnée. Cette estimation a permis d'une part la fixation des paramètres de décision (paramètre de pondération) et d'autre part le choix de l'affectation du fait que les algorithmes proposés fournissent un ensemble de bonnes solutions. Deux approches ont été proposées pour la résolution du problème de décision : une approche d'évaluation multi-critères par Paréto-optimalité et une méthode d'évaluation globale par ε-dominance Paréto-optimalité permettant une minimisation des critères développés.

Dans ce chapitre, nous avons mis l'accent sur quelques axes de recherche qui, a notre avis, présentent un intérêt primordial dans la résolution de problèmes multi-critères. Une approche d'aide à la décision : La recherche d'un ensemble de solutions Paréto-optimales n'est qu'une première étape pour résoudre un PMO. La deuxième étape consiste à faire un choix final parmi les solutions trouvées. Pour ce faire, nous pensons que les algorithmes, qui ne se basent pas a priori sur des préférences, sont généralement plus intéressants, pour les raisons suivantes :

- les décideurs désirent en général plusieurs alternatives et non pas une "meilleure" solution unique ;
- la difficulté de déterminer a priori des préférences sans la connaissance du problème.

La flexibilité, la réactivité et la facilité d'hybridation des méthodes développées avec

l'algorithme génétique ont permis de le développement d'outils intelligents d'aide à l'ordonnancement et à la résolution et à l'évaluation dans le domaine d'optimisation d'une façon générale.

Conclusion générale

Dans cette thèse, nous nous sommes intéressés au problème d'ordonnancement des ateliers flexibles de type job-shop caractérisé par le fait qu'une opération nécessite exactement une machine, choisie dans un ensemble défini a priori, pour être réalisée. Ce type d'ateliers permet de modéliser de nombreux problèmes d'ordonnancement rencontrés dans le secteur de la production et également dans le secteur des services, où il est souvent nécessaire de déterminer la bonne affectation pour chaque tâche.

Par ailleurs, nous nous sommes intéressés à l'aspect optimisation de la production en tenant compte des impératifs économiques qui jouent aujourd'hui un rôle déterminant dans la conduite de la production industrielle. Il a acquis dans l'entreprise manufacturière une position clef.

Notre contribution à ce sujet touche différents aspects. Nous nous sommes focalisé tout d'abord sur les différentes démarches de résolution du problème du job-shop flexible dans le cas où les interactions entre les critères sont supposées disponibles. Cinq critères ont été abordés : le Makespan, la charge critique, la charge totale, la pénalité de retards/avance et le coût de la production.

Notre objectif étant le développement des approches flexibles pouvant être adaptées pour tenir compte de différentes contraintes et de différents critères, notre contribution concerne le développement de nouvelles méthodes efficaces pour la résolution du job-shop flexible.

Concernant la résolution du problème d'affectation, selon le critère global étudié, nous avons défini une estimation du critère global, pour une solution d'affectation donnée, basée sur le calcul des bornes inférieures des critères étudiés.

Trois types d'approches ont été utilisées : une méthode statique basée sur l'intégrale de Choquet, une méthode approchée basée sur le concept Paréto-optimalité et la troisième méthode, basée sur le concept de ϵ-dominance Paréto-optimalité, les critères à minimiser dépendant du critère d'optimisation global étudié.

En effet, pour le problème de minimisation du Makespan, nous avons cherché à optimiser la charge de la machine critique ainsi que la charge totale des machines.

Pour le problème de minimisation du coût total de production et de la somme des retards, nous avons défini des critères appropriés qui sont la pénalité des retards engendrés par les machines et les produits, la pénalité sur le stockage engendré par la mauvaise estimation de la production, et le coût de fabrication des gammes de produits.

Afin d'évaluer les solutions d'affectation pour le critère global, nous avons développé des bornes inférieures pour tous les critères permettant de donner, pour une affectation donnée, une estimation du critère global étudié et d'évaluer par la suite la solution de séquencement.

A partir des méthodes développées pour résoudre le problème d'affectation et des mesures définies pour évaluer une solution d'affectation (bornes inférieures du critère global), nous sommes parvenu à définir une méthodologie performante permettant de déterminer la bonne affectation pour chaque tâche tout en étant capable d'estimer la valeur du critère global. Cette méthodologie peut être intégrée dans un logiciel pour l'ordonnancement dans les systèmes flexibles de production.

Nous avons par la suite introduit une approche intégrée, basée sur les algorithmes génétiques, améliorant les approches évolutionnistes existant dans la littérature. En effet, les limites des codages proposés jusqu'alors pour le problème de job-shop flexible sont de deux natures : la nécessité d'appliquer une routine de correction afin de respecter les contraintes de précédence et/ou la difficulté d'intégrer le séquencement, d'appliquer des opérateurs agissant sur l'ordre de passage des opérations sur les machines et d'explorer les deux espaces de recherche (affectation et séquencement). Notre apport porte dans ce sens sur le développement d'une approche basée sur les algorithmes génétiques, motivé par le fait que ces der-

niers ont pour avantages de déterminer rapidement plusieurs bonnes solutions à un problème d'ordonnancement et de pouvoir être adaptés pour tenir compte de différentes contraintes.

Les approches proposées ont été validées par les bornes inférieures développées et par la comparaison avec d'autres approches de la littérature sur des instances classiques et ont conduit à des résultats satisfaisants.

Par ailleurs, ces approches sont flexibles, génériques et adaptées pour tenir compte des contraintes du problème traité.

En effet, dans un premier temps, nous avons traité le problème de décision et d'évaluation d'une solution. Une approche multi-critères basée sur les intégrales floues, a été développée Pour l'évaluation, les critères appropriés étant basés sur le calcul des bornes inférieures (du Makespan, de la charge critique, de la charge totale, de pénalités de retards et de stockage, et de coût de la production) en présence de contraintes de disponibilité des ressources. Le calcul des bornes inférieures est basé sur la relaxation de la contrainte de non-préemption. Une adaptation d'un algorithme génétique a été proposée pour résoudre le problème d'évaluation des solutions générées en tenant des critères définis, des contraintes de disponibilité sur les machines et des dates de débuts au plus tôt.

Nous nous sommes intéressés par la suite à l'évaluation dynamique multi-critères par Paréto-optimalité. Cet algorithme a été utilisé pour définir une approche de résolution du problème général. Une adaptation de l'approche intégrée a été proposée, pour le job-shop flexible avec contraintes de disponibilité, au niveau de l'agrégation des critères et de l'évaluation des distances entre les valeurs des critères et les bornes inférieures développées. En effet, nous nous sommes basé sur une approche d'évaluation par ϵ-dominance où l'ordonnancement de la production est optimisé par l'approche développée précédemment avec agrégation des distances.

Ainsi, notre contribution concernant un système d'aide à l'ordonnancement, en tenant compte des coûts de la production, porte sur :
– la définition des critères et le développement des bornes pour ces derniers,
– le développement de différentes approches de résolution et d'évaluation pour les

problèmes de grandes tailles à base de génétique, tout en montrant l'efficacité de ces approches en les comparant avec nos bornes inférieures,
- la proposition d'algorithmes permettant de résoudre les problèmes d'atelier flexible d'une manière polynômiale,
- l'étude du cas où l'agrégation des critères est statique ainsi que du cas d'agrégation dynamique facilitant la prise de décision.

Pour ce qui est des nombreuses perspectives possibles de nos travaux, il est intéressant, tout d'abord, d'introduire des améliorations sur les algorithmes proposés en développant des propriétés supplémentaires ; l'approche de résolution du problème général basée sur ces algorithmes serait aussi implementée sur les algorithmes d'optimisation par colonie de fourmis.

Nous pensons par ailleurs que les bornes inférieures développées et l'idée de les utiliser pour évaluer une solution d'affectation peuvent servir pour l'étude des systèmes de production flexibles dans un environnement incertain en basant sur la détermination d'une estimation du critère global à travers le calcul d'une borne inférieure.

En ce qui concerne les tâches de maintenance, il est important d'intégrer le cas de la maintenance périodique permettant la résolution conjointe de la planification de la production et de la maintenance.

Compte tenu que les problèmes d'ordonnancement statique reste loin de la réalité industrielle, et que la considération des données dynamiques et des événements aléatoires (tels que les pannes non envisagées) permet d'appréhender des situations réelles en entreprise, si une machine tombe en panne dans le cas du problème de job-shop flexible, il devient d'abord possible de changer dynamiquement l'affectation des opérations s'exécutant sur cette machine et de réduire ainsi l'impact de la panne et de mettre au point ensuite des méthodes robustes d'ordonnancement conduisant à des solutions satisfaisantes pour lesquelles l'impact de la panne est minimisé.

Bibliographie

[Baccouche, 1995] Baccouche, L. (1995). *Un Mécanisme d'Ordonnancement Distribué de Tâches Temps Réel*. Thèse de Doctorat de l'Institut National Polytechnique de Grenoble, Grenoble.

[Backer, 1974] Backer, K. (1974). *Introduction to sequencing and scheduling*. John Wiley & Sons, New York.

[Berkoune et al., 2005] Berkoune, D., Mesghouni, K., et Rabenasolo, B. (2005). Approche d'évaluation pour des problèmes d'ordonnancement multi-critères : Méthode d'agrégation avec direction de recherche dynamique. *Méthodologies et Heuristiques pour l'Optimisation des Systèmes Industriels, MHOSI'2005, Hammamet*.

[Blazewicz et al., 1996] Blazewicz, J., Ecker, K., Pesch, E., Schmidt, G., et Weglarz, J. (1996). *Scheduling Computer and Manufacturing Processes*. Springer, Berlin.

[Borne et al., 1990] Borne, P., Tanguy, G. D., Richard, J. P., Rotella, F., et Zambettakis, I. (1990). *Commande et optimisation de processus*. Edition Technip, Paris.

[Bouchon-Meunier, 1995] Bouchon-Meunier, B. (1995). *La logique floue et ses applications*. Addison-Wesley, Paris.

[Brucker, 1998] Brucker, P. (1998). *Scheduling Algorithms*. Springer-Verlag, Berlin Heidelberg.

[Carlier et Chretienne, 1988] Carlier, J. et Chretienne, P. (1988). *Problème d'ordonnancement : Modélisation, Complexité, Algorithmes*. Edition Masson, Paris.

[Carlier et al., 1984] Carlier, J., Chrétienne, P., et Girault, C. (1984). *Modelling scheduling problems with Petri nets. Advances studies in Petri nets.* Lecture notes in Computer Science, Springer Verlag, Paris.

[Chu et Proth, 1996] Chu, C. et Proth, J. (1996). *L'ordonnancement et ses applications.* Sciences de l'Ingénieur, Edition Masson, Paris.

[Collette et Siarry, 2002] Collette, Y. et Siarry, P. (2002). *Optimisation multiobjectif.* Editions Eyrolles, Paris.

[Conway et al., 1967] Conway, R., Maxwell, W., et Miller, L. (1967). *Theory of scheduling.* Addison Wesley, Reading, Massachussets.

[Davis, 1985] Davis, L. (1985). Job-shop scheduling with genetic algorithms. *In 1st Int. conf. on Genetic Algorithms and their applications, Lawrence Erl-baum (Ed.), Hillsdale, New Jersey*, pages 136–140.

[Gargouri, 2003] Gargouri, E. (2003). *Ordonnancement Coopératif en Industrie Agroalimentaire.* Thèse de Doctorat de l'Université des Sciences et Technologiques de Lille, Lille.

[Gargouri et al., 1999] Gargouri, E., Hammadi, S., et Borne, P. (1999). Study of scheduling problems in agro-food chain. *Int. Journal of Math. and Computers in Simulation, Intelligent Forecasting, Fault Diagnosis, Scheduling and Control*, 60 :277–291.

[Ghedjati, 1994] Ghedjati, F. (1994). *Résolution par des heuristiques dynamiques et des algorithmes génétiques du problème d'ordonnancement de type job-shop généralisé.* Thèse de Doctorat de l'Université de Paris VI, Paris.

[Glover, 1989] Glover, F. (1989). Tabu search, part i. *ORSA Journal on Computing*, 1(3) :190–206.

[Glover, 1990] Glover, F. (1990). Tabu search, part ii. *ORSA Journal on Computing*, 2(1) :4–32.

[Glover et Laguna, 1997] Glover, F. et Laguna, M. (1997). Tabu search. *Kluwer Academic Publishers.*

[Goldberg, 1994] Goldberg, D. E. (1994). *Algorithmes génétiques : Exploration, optimisation et apprentissage automatique*. Editions Addison-Wesley, S.A., Paris.

[GOTHA, 1993] GOTHA (1993). Les problèmes d'ordonnancement. *RAIRO-Recherche Opérationnelle*, 27(1) :77–150.

[Grabisch, 1996] Grabisch, M. (1996). The application of fuzzy integrals in multicriteria decision making. *European J. of Operational Research*, 89 :445–456.

[Hao et al., 1999] Hao, J., Galinier, P., et Habib, M. (1999). Métaheuristiques pour l'optimisation combinatoire et l'affectation sous contraintes. *Revue d'Intelligence Artificielle, Hermes, Paris*, 13(2) :283–324.

[Kacem, 2003] Kacem, I. (2003). *Ordonnancement multicritère des job-shops flexibles : formulation, bornes inférieures et approche évolutionniste coopérative*. Thèse de Doctorat de l'Université des Sciences et Technologiques de Lille, Lille.

[Kacem et al., 2001] Kacem, I., Hammadi, S., et Borne, P. (2001). Approche évolutionniste modulaire controlée pour le problème du type job-shop flexible. *Journées Doctorales d'Automatique, JDA'2001, Toulouse, 25-27 septembre*.

[Kacem et al., 2002] Kacem, I., Hammadi, S., et Borne., P. (2002). Pareto-optimality approach for flexible job-shop scheduling problems : Hybridization of evolutionary algorithms and fuzzy logic. *Mathematics and Computers in Simulation*, 60 :245–276.

[Kelman, 1996] Kelman, A. (1996). *Modèles flous pour l'agrégation de données et l'aide à la décision*. Thèse de Doctorat de l'Université Paris 6, Paris.

[Kobayashi et al., 1995] Kobayashi, S., Ono, I., et Yamamura, M. (1995). An efficient genetic algorithm for job-shop scheduling problems. *Proceeding of 6th International Conference on Genetic Algorithms*, pages 506–511.

[Laquerbe et al., 1998] Laquerbe, C., Pibouleau, L., Floquet, P., et Domenech, S. (1998). Procedures stochastiques en génie des procédés : Méthode de recuit simulé et algoritthmes génétiques. *6éme Colloque Maghrébin sur les Modèles Numériques de l'Ingénieur, C2MNI6, Tunis*, 12 :761–766.

[Liouane, 1998] Liouane, N. (1998). *Contribution à l'élaboration d'un outil d'aide à la décision pour l'ordonnancement de production en environnement incertain*. Thèse de Doctorat de l'Université des Sciences et Technologiques de Lille, Lille.

[Liu et al., 1998] Liu, X., Begg, D., et Fishwick, R. J. (1998). Genetic approach to optimal topology/controller design of adaptive structures. *In 10th Int. Conf. on Multiple Criteria Decision Making*, 41 :815–830.

[Lopez et Esquirol, 1999] Lopez, P. et Esquirol, P. (1999). *L'ordonnancement*. Edition Economica, Paris.

[Lopez et Roubellat, 2001] Lopez, P. et Roubellat, F. (2001). *Ordonnancement de la production*. Hermes Sciences , IC2 Productique, Paris.

[Mesghouni, 1999] Mesghouni, K. (1999). *Application des algorithmes évolutionnistes dans les problèmes d'optimisation en ordonnancement de la production*. Thèse de Doctorat de l'Université des Sciences et Technologiques de Lille, Lille.

[Mousseau, 1992] Mousseau, V. (1992). Analyse et classification de la littérature traitant de l'importance relative des critères en aide multi-critères à la décision. *Operations Research*, 26(4) :367–389.

[Pinedo, 1955] Pinedo, M. (1955). *Scheduling : Theory, Algorithms and Systems*. Prentice-Hall, Englewood Clis, New Jersey.

[Portmann, 1996] Portmann, M. (1996). Genetic algorithms and scheduling : a state of art and some propositions. *In proceedings Proceedings of the workshop on production planning and control, Mons*.

[Pratyush et Yang, 1998] Pratyush, S. et Yang, J.-B. (1998). *The analytic hierarchy process*. Springer-Verlag, London Limited.

[Rebaudengo et Reorda, 1996] Rebaudengo, M. et Reorda, M. S. (1996). Gallo : a genetic algorithm for floorplan area optimization. *IEEE Transactions on Computer-Aided Design*, 15(8) :991–1000.

[Renders, 1995] Renders, J. M. (1995). *Algorithmes génétiques et Réseaux de Neurones*. Editions Hermes, Paris.

[Roy, 1970] Roy, B. (1970). *Algèbre moderne et théorie des graphes*, volume 2. Dunod, Paris.

[Roy et Bouyssou, 1993] Roy, B. et Bouyssou, D. (1993). *Aide Multi-critères à la décision : Méthodes et cas*. Collection Gestion Série : Production et technologie quantitative appliquées à la gestion. Edition Economica, Paris.

[Saad et Benrejeb, 2006] Saad, I. et Benrejeb, M. (2006). Optimisation multi-critères par paréto-optimale de problèmes d'ordonnancement tenant compte du coût de la production. *Revue Sciences et Technologies de l'Automatique, e-STA*, 3(1).

[Saad et al., 2007a] Saad, I., Boukef, H., et Borne, P. (2007a). The comparative of criteria aggregative approach for the multiobjective optimization flexible job-shop scheduling problems. *Fourth conference on Management and Control of Production and Logistics, MCPL2007, Sibiu, 27-30 September (soumis)*.

[Saad et al., 2007b] Saad, I., Hammadi, S., Benrejeb, M., et Borne, P. (2007b). Choquet integral for criteria aggregation in the flexible job-shop scheduling problems. *Mathematics and Computers in Simulation (MATCOM) (à paraitre)*.

[Saad et al., 2006] Saad, I., Hammadi, S., Borne, P., et Benrejeb, M. (2006). Aggregative approach for the multiobjective optimization flexible job-shop scheduling problems. *the IEEE International Conference on Service Systems and Service Management, IEEE-ICSSSM'06, Troyes, 25-27 october*, pages 889–894.

[Saaty, 1980] Saaty, T. L. (1980). *The analytic hierarchy process*. MacGraw Hill, London.

[Sakarovitch, 1984] Sakarovitch, M. (1984). *Optimisation combinatoire : Méthodes mathématiques et algorithmiques*. Edition Hermann, Paris.

[Serafini, 1992] Serafini, P. (1992). Simulated annealing for multiple objective optimization problems. *In 10th Int. Conf. on Multiple Criteria Decision Making*, pages 87–96.

[Sidi, 2006] Sidi, M. M. O. (2006). *Contribution à l'amélioration des systèmes d'aide à la décision pour la régulation du trafic des réseaux de transport collectif*. Thèse de Doctorat de l'Université des Sciences et Technologiques de Lille, Lille.

[Srinivas et Deb, 1995] Srinivas, N. et Deb, K. (1995). Multiobjective optimisation using non-dominated sorting in genetic algorithms. *Evolutionary Computation*, 2(8) :221–248.

[Syswarda, 1989] Syswarda, G. (1989). Uniform cross-over in genetic algorithms. *Proceedings of Third International Conference on Genetic Algorithms*, pages 2–9.

[Talbi, 1999] Talbi, E. (1999). Métaheuristiques pour l'optimisation combinatoire multiobjectif. *Tutorial, Journées Evolutionnaires Trimestrielles, Paris.*

[Tamaki, 1992] Tamaki, H. (1992). Maintenance of diversity in a genetic algorithm and application to the job-shop scheduling. *Proceedings IMACS/SICE Int. Symp. on MRP2*, pages 869–869.

[Tangour et Saad, 2006] Tangour, F. et Saad, I. (2006). Multiobjective optimization scheduling problems by pareto-optimality in agro-alimentary workshop. *International Journal of Computers, Communications and Control, IJCCC*, I(3) :71–83.

[Xia et Wu, 2005] Xia, W. et Wu, Z. (2005). An effective hybrid optimization approach for multi-objective flexible job-shop scheduling problems. *Computers and Industrial Engineering*, 48(2) :409–425.

[Yager, 1981] Yager, R. (1981). A new methodology for ordinal multi-objective decision based on fuzzy sets. *Decision Sciences*, 12 :589–600.

[Yager, 1988] Yager, R. R. (1988). On weighted median aggregation operators in multicriteria decision making. *IEEE Trans. on Systems, Man and Cybernetics*, 18 :183–190.

[Yamada et Nakano, 1992] Yamada, T. et Nakano, R. (1992). A genetic algorithm applied to large-scale job-shop problems. *In R. Männer et al (Eds.) Parallel Problem Solving From Nature, Amsterdam*, 2 :281–290.

[Yamada et Nakano, 1997] Yamada, T. et Nakano, R. (1997). Genetic algorithms for job-shop scheduling problems. *Proceeding of Modern Heuristic for Decision Support, UNICOM, Seminar, London*, pages 67–81.

[Zitzler et Thiele, 1999] Zitzler, E. et Thiele, L. (1999). Multiobjective evolutionary algorithms : A comparative case study and the strength pareto approach. *IEEE Transactions on Evolutionary Computation*, 3(4) :257–271.

Annexe A

benchmarks

Tab. A.1 – Durées opératoires relatives au benchmark 4×5

		M_1	M_2	M_3	M_4	M_5
J_1	$O_{1,1}$	2	5	4	1	2
	$O_{2,1}$	5	4	5	7	5
	$O_{3,1}$	4	5	5	4	5
J_2	$O_{1,2}$	2	5	4	7	8
	$O_{2,2}$	5	6	9	8	5
	$O_{3,2}$	4	5	4	54	5
J_3	$O_{1,3}$	9	8	6	7	9
	$O_{2,3}$	6	1	2	5	4
	$O_{3,3}$	2	5	4	2	4
	$O_{4,3}$	4	5	2	1	5
J_4	$O_{1,4}$	1	5	2	4	12
	$O_{2,4}$	5	1	2	1	2

Tab. A.2 – Durées opératoires relatives au benchmark 3×6

		M_1	M_2	M_3	M_4	M_5	M_6
J_1	$O_{1,1}$	10	7	6	13	5	1
	$O_{2,1}$	4	5	8	12	7	11
	$O_{3,1}$	9	5	6	12	6	17
	$O_{4,1}$	7	8	4	10	15	3
J_2	$O_{1,2}$	15	12	8	6	10	9
	$O_{2,2}$	9	5	7	13	4	7
	$O_{3,2}$	14	13	14	20	8	17
J_3	$O_{1,3}$	7	16	5	11	17	9
	$O_{2,3}$	9	16	8	11	6	3
	$O_{3,3}$	6	14	8	18	21	14

Tab. A.3 – Durées opératoires relatives au benchmark 8×8

		M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8
J_1	$O_{1,1}$	1	4	6	9	3	5	2	8
	$O_{2,1}$	4	1	1	3	4	8	10	4
	$O_{3,1}$	3	2	5	1	5	6	9	5
J_2	$O_{1,2}$	2	10	4	5	9	8	4	15
	$O_{2,2}$	4	8	7	1	9	6	1	10
	$O_{3,2}$	6	11	2	7	5	3	5	14
	$O_{4,2}$	10	8	9	6	4	7	20	20
J_3	$O_{1,3}$	8	5	8	9	4	3	5	3
	$O_{2,3}$	9	3	6	1	2	6	4	1
	$O_{3,3}$	7	1	8	5	4	9	1	2
J_4	$O_{1,4}$	5	10	6	4	9	5	1	7
	$O_{2,4}$	4	2	3	8	7	4	6	9
	$O_{3,4}$	7	3	12	1	6	5	8	3
J_5	$O_{1,5}$	7	10	4	5	6	3	5	15
	$O_{2,5}$	5	6	3	9	8	2	8	6
	$O_{3,5}$	6	1	4	1	10	4	3	11
	$O_{4,5}$	11	9	20	6	7	5	3	6
J_6	$O_{1,6}$	8	9	10	8	4	2	7	8
	$O_{2,6}$	7	3	12	5	4	3	6	9
	$O_{3,6}$	4	7	3	6	3	4	1	5
J_7	$O_{1,7}$	1	7	8	3	4	9	4	13
	$O_{2,7}$	3	8	1	2	3	6	11	2
	$O_{3,7}$	5	4	2	1	2	1	8	14
J_8	$O_{1,8}$	5	7	11	3	2	9	8	5
	$O_{2,8}$	8	3	10	7	5	13	4	6
	$O_{3,8}$	6	2	13	5	4	3	5	7
	$O_{4,8}$	9	20	3	7	1	5	8	20

Tab. A.4 – Durées opératoires relatives au benchmark 10×10

		M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
J_1	$O_{1,1}$	1	4	6	9	3	5	2	8	9	5
	$O_{2,1}$	4	1	1	3	4	8	10	4	11	4
	$O_{3,1}$	3	2	5	1	5	6	9	5	10	3
J_2	$O_{1,2}$	2	10	4	5	9	8	4	15	8	4
	$O_{2,2}$	4	8	7	1	9	6	1	10	7	1
	$O_{3,2}$	6	11	2	7	5	3	5	14	9	2
J_3	$O_{1,3}$	8	5	8	9	4	3	5	3	8	1
	$O_{2,3}$	9	3	6	1	2	6	4	1	7	2
	$O_{3,3}$	7	1	8	5	4	9	1	2	3	4
J_4	$O_{1,4}$	5	10	6	4	9	5	1	7	1	6
	$O_{2,4}$	4	2	3	8	7	4	6	9	8	4
	$O_{3,4}$	7	3	12	1	6	5	8	3	5	2
J_5	$O_{1,5}$	7	10	4	5	6	3	5	15	2	6
	$O_{2,5}$	5	6	3	9	8	2	8	6	1	7
	$O_{3,5}$	6	1	4	1	10	4	3	11	13	9
J_6	$O_{1,6}$	8	9	10	8	4	2	7	8	3	10
	$O_{2,6}$	7	3	12	5	4	3	6	9	2	15
	$O_{3,6}$	4	7	3	6	3	4	1	5	1	11
J_7	$O_{1,7}$	1	7	8	3	4	9	4	13	10	7
	$O_{2,7}$	3	8	1	2	3	6	11	2	13	3
	$O_{3,7}$	5	4	2	1	2	1	8	14	5	7
J_8	$O_{1,8}$	5	7	11	3	2	9	8	5	12	8
	$O_{2,8}$	8	3	10	7	5	13	4	6	8	4
	$O_{3,8}$	6	2	13	5	4	3	5	7	9	5
J_9	$O_{1,9}$	3	9	1	3	8	1	6	7	5	4
	$O_{2,9}$	4	6	2	5	7	3	1	9	6	7
	$O_{3,9}$	8	5	4	8	6	1	2	3	10	12
J_{10}	$O_{1,10}$	4	3	1	6	7	1	2	6	20	6
	$O_{2,10}$	3	1	8	1	9	4	1	4	17	15
	$O_{3,10}$	9	2	4	2	3	5	2	4	10	23

Tab. A.5 – Durées opératoires relatives au benchmark 10×7

		M_1	M_2	M_3	M_4	M_5	M_6	M_7
J_1	$O_{1,1}$	1	4	6	9	3	5	2
	$O_{2,1}$	8	9	5	4	1	1	3
	$O_{3,1}$	4	8	10	4	11	4	3
J_2	$O_{1,2}$	6	9	8	6	5	10	3
	$O_{2,2}$	2	10	4	5	9	8	4
J_3	$O_{1,3}$	15	4	8	4	8	7	1
	$O_{2,3}$	9	6	1	10	7	1	6
	$O_{3,3}$	11	2	7	5	2	3	14
J_4	$O_{1,4}$	2	8	5	8	9	4	3
	$O_{2,4}$	5	3	8	1	9	3	6
	$O_{3,4}$	1	2	6	4	1	7	2
J_5	$O_{1,5}$	7	1	8	5	1	3	9
	$O_{2,5}$	2	4	5	10	6	4	9
	$O_{3,5}$	5	1	7	1	6	6	2
J_6	$O_{1,6}$	8	7	4	56	9	8	4
	$O_{2,6}$	5	14	1	9	6	5	8
	$O_{3,6}$	3	5	2	5	4	5	7
J_7	$O_{1,7}$	5	6	3	6	5	15	2
	$O_{2,7}$	6	5	4	9	5	4	3
	$O_{3,7}$	9	8	2	8	6	1	7
J_8	$O_{1,8}$	6	1	4	1	10	4	3
	$O_{2,8}$	11	13	9	8	9	10	8
	$O_{3,8}$	4	2	7	8	3	10	7
J_9	$O_{1,9}$	12	5	4	5	4	5	5
	$O_{2,9}$	4	2	15	99	4	7	3
	$O_{3,9}$	9	5	11	2	5	4	2
J_{10}	$O_{1,10}$	9	4	13	10	7	6	8
	$O_{2,10}$	4	3	25	3	8	1	2
	$O_{3,10}$	1	2	6	11	13	3	5

Tab. A.6 – Durées opératoires relatives au benchmark 15×10

		M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
J_1	$O_{1,1}$	1	4	6	9	3	5	2	8	9	4
	$O_{2,1}$	1	1	3	4	8	10	4	11	4	3
	$O_{3,1}$	2	5	1	5	6	9	5	10	3	2
	$O_{4,1}$	10	4	5	9	8	4	15	8	4	4
J_2	$O_{1,2}$	4	8	7	1	9	6	1	10	7	1
	$O_{2,2}$	6	11	2	7	5	3	5	14	9	2
	$O_{3,2}$	8	5	8	9	4	3	5	3	8	1
	$O_{4,2}$	9	3	6	1	2	6	4	1	7	2
J_3	$O_{1,3}$	7	1	8	5	4	9	1	2	3	4
	$O_{2,3}$	5	10	6	4	9	5	1	7	1	6
	$O_{3,3}$	4	2	3	8	7	4	6	9	8	4
	$O_{4,3}$	7	3	12	1	6	5	8	3	5	2
J_4	$O_{1,4}$	6	2	5	4	1	2	3	6	5	4
	$O_{2,4}$	8	5	7	4	1	2	36	5	8	5
	$O_{3,4}$	9	6	2	4	5	1	3	6	5	2
	$O_{4,4}$	11	4	5	6	2	7	5	4	2	1
J_5	$O_{1,5}$	6	2	5	4	1	2	3	6	5	4
	$O_{2,5}$	5	4	6	3	5	2	28	7	4	5
	$O_{3,5}$	6	2	4	3	6	5	2	4	7	9
	$O_{4,5}$	6	5	4	2	3	2	5	4	7	5
J_6	$O_{1,6}$	4	1	3	2	6	9	8	5	4	2
	$O_{2,6}$	1	3	6	5	4	7	5	4	6	5
J_7	$O_{1,7}$	1	4	2	5	3	6	9	8	5	4
	$O_{2,7}$	2	1	4	5	2	3	5	4	2	5
J_8	$O_{1,8}$	2	3	6	2	5	4	1	5	8	7
	$O_{2,8}$	4	5	6	2	3	5	4	1	2	5
	$O_{3,8}$	3	5	4	2	5	49	8	5	4	5
	$O_{4,8}$	1	2	36	5	2	3	6	4	11	2
J_9	$O_{1,9}$	6	3	2	22	44	11	10	23	5	1
	$O_{2,9}$	2	3	2	12	15	10	12	14	18	16
	$O_{3,9}$	20	17	12	5	9	6	4	7	5	6
	$O_{4,9}$	9	8	7	4	5	8	7	4	56	2
J_{10}	$O_{1,10}$	5	8	7	4	56	3	2	5	4	1
	$O_{2,10}$	2	5	6	9	8	5	4	2	5	4
	$O_{3,10}$	6	3	2	5	4	7	4	5	2	1
	$O_{4,10}$	3	2	5	6	5	8	7	4	5	2
J_{11}	$O_{1,11}$	1	2	3	6	5	2	1	4	2	1
	$O_{2,11}$	2	3	6	3	2	1	4	10	12	1
	$O_{3,11}$	3	6	2	5	8	4	6	3	2	5
	$O_{4,11}$	4	1	45	6	2	4	1	25	2	4
J_{12}	$O_{1,12}$	9	8	5	6	3	6	5	2	4	2
	$O_{2,12}$	5	8	9	5	4	75	63	6	5	21
	$O_{3,12}$	12	5	4	6	3	2	5	4	2	5
	$O_{4,12}$	8	7	9	5	6	3	2	5	8	4
J_{13}	$O_{1,13}$	4	2	5	6	8	5	6	4	6	2
	$O_{2,13}$	3	5	4	7	5	8	6	6	3	2
	$O_{3,13}$	5	4	5	8	5	4	6	5	4	2
	$O_{4,13}$	3	2	5	6	5	4	8	5	6	4
J_{14}	$O_{1,14}$	2	3	5	4	6	5	4	85	4	5
	$O_{2,14}$	6	2	4	5	8	6	5	4	2	6
	$O_{3,14}$	3	25	4	8	5	6	3	2	5	4
	$O_{4,14}$	8	5	6	4	2	3	6	8	5	4
J_{15}	$O_{1,15}$	2	5	6	8	5	6	3	2	5	4
	$O_{2,15}$	5	6	2	5	4	2	5	3	2	5
	$O_{3,15}$	4	5	2	3	5	2	8	4	7	5
	$O_{4,15}$	6	2	11	14	2	3	6	5	4	8

Tab. A.7 – Durées opératoires relatives au benchmark 9×8

		M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8
J_1	$O_{1,1}$	1	4	6	9	3	5	2	8
	$O_{2,1}$	4	1	1	3	4	8	10	4
	$O_{3,1}$	3	2	5	1	5	6	9	5
J_2	$O_{1,2}$	2	10	4	5	9	8	4	15
	$O_{2,2}$	4	8	7	1	9	6	1	10
	$O_{3,2}$	6	11	2	7	5	3	5	14
J_3	$O_{1,3}$	8	5	8	9	4	3	5	3
	$O_{2,3}$	9	3	6	1	2	6	4	1
	$O_{3,3}$	7	1	8	5	4	9	1	2
J_4	$O_{1,4}$	5	10	6	4	9	5	1	7
	$O_{2,4}$	4	2	3	8	7	4	6	9
	$O_{3,4}$	7	3	12	1	6	5	8	3
J_5	$O_{1,5}$	7	10	4	5	6	3	5	15
	$O_{2,5}$	5	6	3	9	8	2	8	6
	$O_{3,5}$	6	1	4	1	10	4	3	11
J_6	$O_{1,6}$	8	9	10	8	4	2	7	8
	$O_{2,6}$	7	3	12	5	4	3	6	9
	$O_{3,6}$	4	7	3	6	3	4	1	5
J_7	$O_{1,7}$	1	7	8	3	4	9	4	13
	$O_{2,7}$	3	8	1	2	3	6	11	2
	$O_{3,7}$	5	4	2	1	2	1	8	14
J_8	$O_{1,8}$	5	7	11	3	2	9	8	5
	$O_{2,8}$	8	3	10	7	5	13	4	6
	$O_{3,8}$	6	2	13	5	4	3	5	7
J_9	$O_{1,9}$	11	9	20	6	7	5	3	6
	$O_{2,9}$	9	20	3	7	1	5	8	20
	$O_{3,9}$	10	8	9	6	4	7	20	20

TAB. A.8 – Durées opératoires relatives au benchmark 12×9

		M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9
J_1	$O_{1,1}$	1	4	6	9	3	5	2	8	9
	$O_{2,1}$	1	1	3	4	8	10	4	11	4
	$O_{3,1}$	2	5	1	5	6	9	5	10	3
J_2	$O_{1,2}$	4	8	7	1	9	6	1	10	7
	$O_{2,2}$	6	11	2	7	5	3	5	14	9
	$O_{3,2}$	8	5	8	9	4	3	5	3	8
J_3	$O_{1,3}$	7	1	8	5	4	9	1	2	3
	$O_{2,3}$	5	10	6	4	9	5	1	7	1
	$O_{3,3}$	4	2	3	8	7	4	6	9	8
J_4	$O_{1,4}$	6	2	5	4	1	2	3	6	5
	$O_{2,4}$	8	5	7	4	1	2	36	5	8
	$O_{3,4}$	9	6	2	4	5	1	3	6	5
J_5	$O_{1,5}$	6	2	5	4	1	2	3	6	5
	$O_{2,5}$	5	4	6	3	5	2	28	7	4
	$O_{3,5}$	6	2	4	3	6	5	2	4	7
J_6	$O_{1,6}$	4	1	3	2	6	9	8	5	4
	$O_{2,6}$	1	3	6	5	4	7	5	4	6
	$O_{3,6}$	10	4	5	9	8	4	15	8	4
J_7	$O_{1,7}$	1	4	2	5	3	6	9	8	5
	$O_{2,7}$	2	1	4	5	2	3	5	4	2
	$O_{3,7}$	6	5	4	2	3	2	5	4	7
J_8	$O_{1,8}$	2	3	6	2	5	4	1	5	8
	$O_{2,8}$	4	5	6	2	3	5	4	1	2
	$O_{3,8}$	3	5	4	2	5	49	8	5	4
J_9	$O_{1,9}$	6	3	2	22	44	11	10	23	5
	$O_{2,9}$	2	3	2	12	15	10	12	14	18
	$O_{3,9}$	20	17	12	5	9	6	4	7	5
J_{10}	$O_{1,10}$	5	8	7	4	56	3	2	5	4
	$O_{2,10}$	2	5	6	9	8	5	4	2	5
	$O_{3,10}$	6	3	2	5	4	7	4	5	2
J_{11}	$O_{1,11}$	1	2	3	6	5	2	1	4	2
	$O_{2,11}$	2	3	6	3	2	1	4	10	12
	$O_{3,11}$	3	6	2	5	8	4	6	3	2
J_{12}	$O_{1,12}$	9	8	5	6	3	6	5	2	4
	$O_{2,12}$	5	8	9	5	4	75	63	6	5
	$O_{3,12}$	12	5	4	6	3	2	5	4	2

Tab. A.9 – Durées opératoires relatives au benchmark 20×10

		M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
J_1	$O_{1,1}$	1	4	6	9	3	5	2	8	9	4
	$O_{2,1}$	1	1	3	4	8	10	4	11	4	3
	$O_{3,1}$	2	5	1	5	6	9	5	10	3	2
J_2	$O_{1,2}$	4	8	7	1	9	6	1	10	7	1
	$O_{2,2}$	6	11	2	7	5	3	5	14	9	2
	$O_{3,2}$	8	5	8	9	4	3	5	3	8	1
J_3	$O_{1,3}$	7	1	8	5	4	9	1	2	3	4
	$O_{2,3}$	5	10	6	4	9	5	1	7	1	6
	$O_{3,3}$	4	2	3	8	7	4	6	9	8	4
J_4	$O_{1,4}$	6	2	5	4	1	2	3	6	5	4
	$O_{2,4}$	8	5	7	4	1	2	36	5	8	5
	$O_{3,4}$	9	6	2	4	5	1	3	6	5	2
J_5	$O_{1,5}$	6	2	5	4	1	2	3	6	5	4
	$O_{2,5}$	5	4	6	3	5	2	28	7	4	5
	$O_{3,5}$	6	2	4	3	6	5	2	4	7	9
J_6	$O_{1,6}$	4	1	3	2	6	9	8	5	4	2
	$O_{2,6}$	1	3	6	5	4	7	5	4	6	5
	$O_{3,6}$	1	2	36	5	2	3	6	4	11	2
J_7	$O_{1,7}$	1	4	2	5	3	6	9	8	5	4
	$O_{2,7}$	2	1	4	5	2	3	5	4	2	5
	$O_{3,7}$	9	8	7	4	5	8	7	4	56	2
J_8	$O_{1,8}$	2	3	6	2	5	4	1	5	8	7
	$O_{2,8}$	4	5	6	2	3	5	4	1	2	5
	$O_{3,8}$	3	5	4	2	5	49	8	5	4	5
J_9	$O_{1,9}$	6	3	2	22	44	11	10	23	5	1
	$O_{2,9}$	2	3	2	12	15	10	12	14	18	16
	$O_{3,9}$	20	17	12	5	9	6	4	7	5	6
J_{10}	$O_{1,10}$	5	8	7	4	56	3	2	5	4	1
	$O_{2,10}$	2	5	6	9	8	5	4	2	5	4
	$O_{3,10}$	6	3	2	5	4	7	4	5	2	1
J_{11}	$O_{1,11}$	1	2	3	6	5	2	1	4	2	1
	$O_{2,11}$	2	3	6	3	2	1	4	10	12	1
	$O_{3,11}$	3	6	2	5	8	4	6	3	2	5
J_{12}	$O_{1,12}$	9	8	5	6	3	6	5	2	4	2
	$O_{2,12}$	5	8	9	5	4	75	63	6	5	21
	$O_{3,12}$	12	5	4	6	3	2	5	4	2	5
J_{13}	$O_{1,13}$	4	2	5	6	8	5	6	4	6	2
	$O_{2,13}$	3	5	4	7	5	8	6	6	3	2
	$O_{3,13}$	5	4	5	8	5	4	6	5	4	2
J_{14}	$O_{1,14}$	2	3	5	4	6	5	4	85	4	5
	$O_{2,14}$	6	2	4	5	8	6	5	4	2	6
	$O_{3,14}$	3	25	4	8	5	6	3	2	5	4
J_{15}	$O_{1,15}$	2	5	6	8	5	6	3	2	5	4
	$O_{2,15}$	5	6	2	5	4	2	5	3	2	5
	$O_{3,15}$	4	5	2	3	5	2	8	4	7	5
J_16	$O_{1,16}$	8	9	10	8	4	2	7	8	3	10
	$O_{2,16}$	7	3	12	5	4	3	6	9	2	15
	$O_{3,16}$	4	7	3	6	3	4	1	5	1	11
J_17	$O_{1,17}$	1	7	8	3	4	9	4	13	10	7
	$O_{2,17}$	3	8	1	2	3	6	11	2	13	3
	$O_{3,17}$	5	4	2	1	2	1	8	14	5	7
J_18	$O_{1,18}$	5	7	11	3	2	9	8	5	12	8
	$O_{2,18}$	8	3	10	7	5	13	4	6	8	4
	$O_{3,18}$	6	2	13	5	4	3	5	7	9	5
J_19	$O_{1,19}$	3	9	1	3	8	1	6	7	5	4
	$O_{2,19}$	4	6	2	5	7	3	1	9	6	7
	$O_{3,19}$	8	5	4	8	6	1	2	3	10	12
J_{20}	$O_{1,20}$	4	3	1	6	7	1	2	6	20	6
	$O_{2,20}$	3	1	8	1	9	4	1	4	17	15
	$O_{3,20}$	9	2	4	2	3	5	2	4	10	23

Conception d'un système d'aide à l'ordonnancement tenant compte des impératifs économiques

Résumé

Nos travaux concernent la mise en œuvre de méthodologies pour la résolution et l'optimisation de la production en tenant compte des impératifs économiques, jouant aujourd'hui un rôle déterminant dans la conduite de la production industrielle. Pour le problème du job-shop flexible dans lequel les interactions entre les critères sont supposées disponibles, cinq critères ont été retenus : le Makespan, la charge critique, la charge totale, la pénalité de retards/avance et le coût de la production. Dans ce sens, nous avons, d'abord, traité le problème de décision et d'évaluation d'une solution et introduit ensuite trois approches intégrées, basées sur les algorithmes génétiques, améliorant les approches évolutionnistes existant dans la littérature : la méthode statique basée sur l'intégrale de Choquet, la méthode approchée basée sur le concept Paréto-optimalité ainsi que la méthode basée sur le concept de ε-dominance Paréto-optimalité. Les approches adoptées consistent à générer une variété de solutions optimales diversifiées dans l'espace de recherche de solutions, et d'aider le décideur, quand il ne peut pas donner une préférence particulière à l'une des fonctions objectif. Les résultats proposés, obtenus globalement pour l'ensemble des critères, ont été comparés, avec succès, avec ceux obtenus par d'autres approches existantes sur plusieurs benchmarks de complexités distinctes

Mots-clés : job-shop, flexibilité, multi-critères, coût de la production, bornes inférieures, intégrale de Choquet, Paréto-optimalité, dominance, algorithme génétique.

Scheduling support system design taking into account economic imperatives

Abstract

Our works deal with the implementation of methodologies for production resolution and optimization taking into consideration economic imperatives, which are important parts in industrial manufacturing control. For the flexible job-shop problem, five criteria are considered : Makespan, the workload of the critical machine, the total workload of machines, the penalties of earliness/tardiness and the production cost. In this direction, the decision and evaluation problems are introduced and studied, and three integrated approaches, based on genetic algorithms, improving evolutionary approaches which exist in the literature, are presented : static method based on Choquet's integral, approached method based on the Pareto-optimality concept as well as the method based on the concept of ε-dominance Pareto-optimality.

The proposed approaches are used to generate a variety of optimal solutions needed to help the decision-maker, when he can not give a particular preference to one of the objective functions. Proposed results, obtained globally for the overall of criteria, are successfully compared with those obtained by other existing approaches on several benchmarks presenting different complexities.

Key-words : job-shop, flexibility, multi-criteria, production cost, lower bounds, Choquet integral, Pareto-optimality, dominance, genetic algorithm.