Sommaire

Liste de figures

Liste de tables

INTRODUCTION GÉNÉRALE

L'industrie actuelle se caractérise par une forte demande de produits personnalisés de bonne qualité et à bas prix, dans des délais de plus en plus raccourcis. En effet, l'ouverture des marchés internationaux, ainsi que l'évolution et la mondialisation ont poussé les industriels à se diriger vers des systèmes de fabrication de plus en plus flexibles, ce qui a entrainé à la mise en cause de plusieurs habitudes de production et en particulier la gestion des ateliers de production qui joue un rôle crucial dans la productivité et le raccourcissement des délais de production. Un système de production est dit flexible s'il peut assurer la production simultanée de plusieurs types de pièces avec des quantités variables et s'il est capable de s'adapter à la production de nouveaux produits pour lesquels le système n'a pas été étudié (Esquirol et Lopez 1999).

La productivité peut être affectée directement par la qualité de l'ordonnancement des opérations sur les machines, car un atelier de production peut réaliser une grande variété de produits avec des coûts réduits, grâce à une meilleure utilisation des ressources. Le domaine d'application de l'ordonnancement est vaste : par exemple, la gestion de la charge des processus en informatique, la gestion de la production dans l'industrie, la gestion de projets, etc.

Le problème d'ordonnancement est classé parmi les problèmes fortement combinatoires, et il est toujours renouvelable, car jusqu'à maintenant, il n'existe aucune méthode de résolution générale et de faible complexité algorithmique (Tangour 2007).

Nous trouvons une grande variété de problèmes d'ordonnancement qui sont liés à plusieurs paramètres : les tâches ou opérations (préemptives et non préemptives, indépendantes ou non), les paramètres relatifs aux ressources (renouvelables, consommables), types de contraintes sur les tâches (précédence, disjonctions), critères d'optimalité (minimisation de la durée totale d'achèvement de toutes les tâches « *makespan* », minimisation du retard total des tâches, minimisation de la charge des machines, etc.).

La résolution du problème d'ordonnancement consiste à attribuer une ressource à chaque opération et à organiser l'ordre d'exécution des tâches en attribuant une date de début à chaque tâche, de telle sorte à respecter toutes les contraintes considérées dans le but d'optimiser un certain nombre d'objectifs (Drakaki et Tzionas 2017).

Intuitivement, la résolution d'un problème d'ordonnancement passe par deux étapes fondamentales : 1. identifier et modéliser le problème en décrivant les contraintes qui doivent être respectées et en cernant les critères à optimiser. 2. trouver la méthode la plus adéquate pour la résolution du problème considéré. En effet,

l'exploitation de plusieurs méthodes est importante afin d'établir des décisions efficaces (Tangour 2007).

Dans ce travail, nous avons abordé deux problématiques :

- *Problématique 1* : l'ordonnancement des ateliers de type Job Shop flexible (JSF) avec temps de transport,

- *Problématique 2 :* l'ordonnancement du JSF avec temps de transport et contrôle de consommation d'énergie électrique des machines.

Dans les ateliers *JSF*, il existe plusieurs Jobs à exécuter par plusieurs machines. Mais l'achèvement de chaque Job se traduit par l'exécution de plusieurs tâches ou opérations élémentaires (selon un ordre prédéfini), ce qu'on appelle la gamme opératoire de chaque Job. Les opérations dans l'atelier Job Shop peuvent être traitées sur diverses machines. Dans notre cas, il s'agit d'un atelier Job Shop Flexible, c'est-à-dire: il existe plusieurs machines candidates pour chaque opération, de telle sorte que le temps opératoire de chaque opération dépend des machines candidates. Nous ciblons l'efficacité en minimisant le temps de production. Nous visons aussi l'efficience en minimisant l'énergie électrique consommée par les machines durant la production.

Le cas d'étude choisi est la cellule de production AIP-PREMICA (Trentesaux, et al. 2013), développé au niveau de l'université de Valenciennes, en France. Cette cellule reflète un modèle réel du job shop flexible, dont les jobs sont transportés par des convoyeurs.

Nous proposons deux approches pour les problématiques considérées :

- *Approche pour la problématique 1* :

Un algorithme génétique hybride basé sur la recherche tabou. Une fonction de voisinage basée sur les fenêtres temporelles libres et occupées est proposée pour la recherche tabou. La fonction objectif considérée est la durée de production (makespan).

- *Approche pour la problématique 2* :

Une approche prédictive-réactive d'optimisation multi-objectifs en prenant en considération la consommation de l'énergie des machines durant la production. L'approche prédictive est basée sur NSGA-2 (Non Dominant Sorting Genetic Algorithm) avec trois fonctions objectifs : makespan, consommation totale de l'énergie et le pic de consommation de l'énergie. L'algorithme de contrôle d'énergie réagit en cas d'une nécessité pour limiter le pic de consommation de l'énergie. Un modèle pour limiter la perte de l'énergie dans les périodes où les machines ne sont pas utilisées est proposé.

Cette thèse est organisée de la manière suivante :

Dans le chapitre 1, nous introduisons les différents problèmes d'ordonnancement. Ensuite, nous enchainons avec les différents types d'ateliers, en cernant les caractéristiques nécessaires du problème d'ordonnancement, ainsi que sa complexité. Puis, les méthodes de modélisation et de représentation des solutions seront évoquées.

Dans le chapitre 2, après avoir présenté des notions générales sur les problèmes d'optimisation, nous présentons une formulation des problèmes d'optimisation et les différentes méthodes et approches de résolution, à savoir les approches exactes et les approches approximatives.

Dans le chapitre 3, nous illustrons l'approche proposée pour l'optimisation de la durée de production (makespan), tout en prenant en considération le temps de transport entre les différentes machines. L'approche proposée est testée sur la cellule de production AIP- PRIMECA.

Une approche multi-objectifs pour l'efficience de l'énergie est présentée dans le chapitre 4. Contrairement à l'approche mono objectif, proposée dans le chapitre 3, cette approche prend en considération la consommation de l'énergie durant la production. Nous présentons les résultats obtenus en fin de ce chapitre.

Enfin, nous clôturons cette thèse par une conclusion générale et nous donnons quelques perspectives.

NOTATION

- j Indexe du Job à réaliser. $j = 1, \dots, N$.
- i Indexe de l'opération. $i = 1, \dots, a_j$.
- k Indexe de la machine M_k. $k = 1, \dots, M$.
- a_j Dernière opération du Job J_j
- $O_{j,i}$ Opération numéro i appartenant au job j.
- $P_{j,i,k}$ Durée de traitement de l'opération $O_{i,j}$ sur la machine M_k.
- $t_{j,i}$ Temps de début de l'opération $O_{i,j}$.
- $tf_{j,i}$ Temps de fin de l'opération $O_{i,j}$.
- G_j Ensemble d'opérations constituant la gamme opératoire du Job $J_j / G_j = \{O_{1,j}, O_{2,j}, \dots, O_{a_j,j}\}$.
- $X_{j,i,k}$ $X_{j,i,k} = 1$ Si l'opération $O_{j,i}$ est assignée à la machine M_k.

  Sinon $X_{j,i,k} = 0$.

- C_{max} Temps d'achèvement de toutes les Jobs :

  $$C_{max} = \max_{j=1,N} tf_{a_j}.$$

- W La machine la plus chargée /

  $W = \max_{k=1,M} W_k$ / W_k : La somme des durées d'opérations assignées à la machine M_k :

$$W_k = \sum_{i=1}^{a_j} \sum_{j=1}^{N} (P_{i,j,k} * X_{i,j,k}).$$

- $Tr_{kk'}$ le temps de transport de la machine k vers la machine k'.
- Pr_k^{Idle} : le niveau de puissance électrique [1] de la machine M_k en mode veille (Watts).
- $Pr_{j,i,k}^{Assembling}$: le niveau de puissance électrique de la machine M_k durant l'assemblage
- $E_{j,i,k}^{Assembling}$: l'énergie consommée par la machine M_k, pendant l'assemblage ou chargement/déchargement de l'opération $O_{j,i,k}$ (Watts-hour).

[1] La puissance électrique Pe reçue par un appareil électrique est égale au produit de la tension électrique U entre ses bornes et de l'intensité I du courant électrique qui le traverse

CHAPITRE 1 : L'ORDONNANCEMENT ET SES CARACTERISTIQUES DANS LES SYSTEMES DE PRODUCTION

1. INTRODUCTION

Quelque soit le domaine d'industrie, les coûts de production ainsi que les délais de livraison représentent deux fonctions cruciales pour l'efficacité et la compétitivité entre les entreprises. Ces deux fonctions sont étroitement liées à la qualité de l'ordonnancement adopté dans le processus de production, c'est pour cette raison que les problèmes d'ordonnancement sont devenus de plus en plus importants dans le monde des systèmes de production.

En revanche, être efficace lorsque l'environnement est déterministe ne suffit pas et un système doit pouvoir réagir à l'incertitude et aux perturbations (ordre de fabrication urgents, panne des machines, baisse de la quantité d'énergie disponible, etc.).

Dans ce chapitre introductif, nous cherchons dans un premier temps à positionner le concept d'ordonnancement par rapport à la gestion de production, ensuite, nous présenterons les notions de base relatives à l'ordonnancement dans les ateliers de production. La section suivante sera consacrée aux différents types d'ateliers de production. Ensuite, nous présentons la notion de complexité, à savoir la complexité algorithmique et la complexité problématique. Les deux sections suivantes porteront sur la classification et la modélisation des problèmes d'ordonnancement. Ce chapitre sera clôturé par les particularités de l'ordonnancement dans les ateliers Job Shop Flexibles.

2. LES SYSTEMES DE PRODUCTION

La production est le processus conduisant à la création des produits par l'utilisation et la transformation de ressources (Giard 2003). Le processus de production est alors, constitué d'un ensemble d'opérations qui sont les activités conduisant à la création de biens et de services (Javel 2004).

Le système de production est l'ensemble des ressources réalisant une activité de production. C'est un ensemble de moyens divers : humains, matériels, informationnels et autres, constituant un tout, dont l'objectif est la réalisation de biens et de services (Fontanili 1999) .

Un système de production peut être décomposé en trois sous-systèmes (**Figure 0.1**) (Javel 2004):

- Le système physique de production : ce système englobe toutes les ressources humaines et physiques nécessaires pour la transformation des matières premières en produits finis.

- Le système de décision : ce système contrôle le système physique de production à travers l'organisation des différentes activités en prenant des décisions basées sur les données transmises par le système informationnel.

- Le système d'information : d'une part, ce système joue un rôle d'interface entre le système de décision et le système de production et d'autre part, il intervient à l'intérieur du système de décision, pour la gestion des informations utilisées lors de la prise de décision. Son rôle et de collecter, stocker et transmettre des informations de différents types (Fontanili 1999).

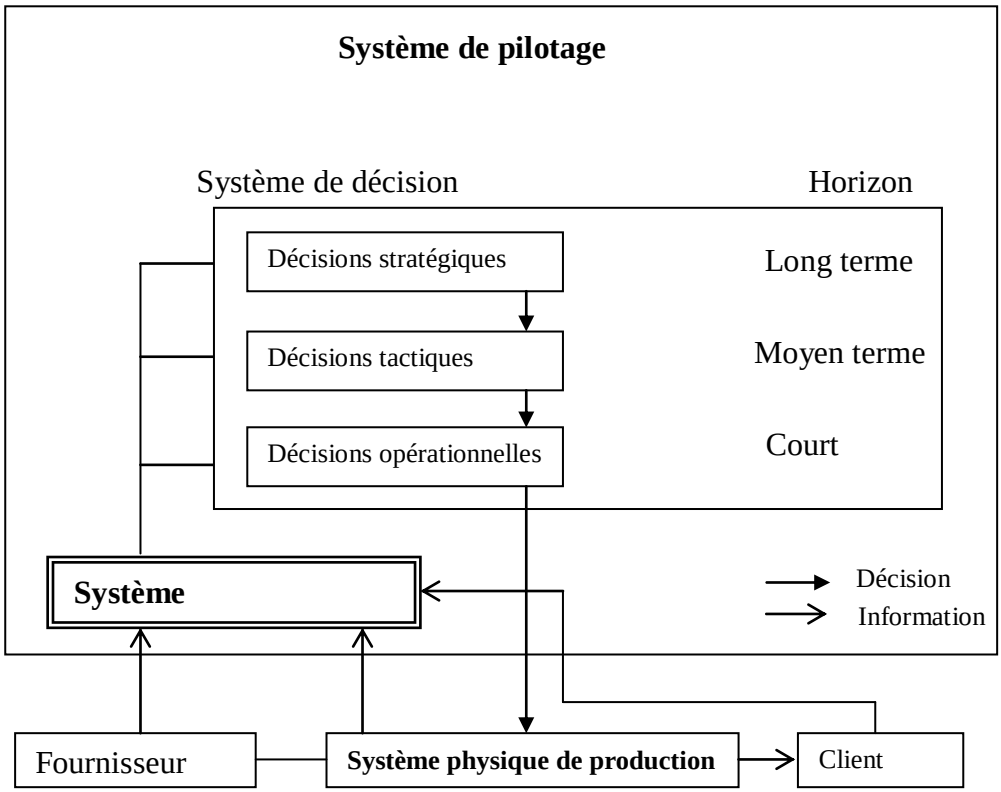


Figure 0.1Décomposition d'un système de production (Javel 2004)

3. LA GESTION DE PRODUCTION

3.1 Définition et rôle de la gestion de production

La gestion de production est « *la fonction qui permet de réaliser les opérations de production en respectant les conditions de qualité, délais, coût qui résultent des objectifs de l'entreprise* » (Blondel 2004).

La gestion de production englobe plusieurs aspects, tels que la planification, la gestion des stocks, la prévision, l'ordonnancement, le contrôle, etc.

3.2 Décomposition hiérarchique de la gestion de production

Nous distinguons trois niveaux hiérarchiques de la gestion de production (**Figure 0.1**) : stratégique, tactique et opérationnel (Giard 2003) (Fontanili 1999):

a. le niveau stratégique :

Ce niveau trace la politique à long terme de l'entreprise (à un horizon de plus de deux ans). Cette politique porte essentiellement sur la gestion des ressources durables, afin que celles-ci soient en mesure d'assurer la pérennité de l'entreprise.

b. le niveau tactique :

Ce niveau relie les deux niveaux stratégique et opérationnel, il porte sur les décisions à moyen terme. Le but est d'assurer une production satisfaisante à la demande en minimisant les coûts, tout en respectant le plan tracé par le niveau stratégique de l'entreprise.

c. le niveau opérationnel :

Ce niveau porte sur les décisions à court terme. Il s'agit d'une gestion quotidienne pour satisfaire les demandes en respectant les décisions tactiques. Parmi les décisions opérationnelles : la gestion de la main d'œuvre, la gestion des stocks, la gestion des équipements (Blondel 2004).

3.3 Rôle de l'ordonnancement dans la gestion de production

Le modèle général en gestion de production, décompose les décisions en trois niveaux :

Stratégique, tactique et opérationnel (pilotage et suivi quotidien des flux de matières et du travail). Cette hiérarchisation se traduit par une échelle de responsabilité (direction, cadre, agent, etc.) (Esquirol et Lopez 1999)

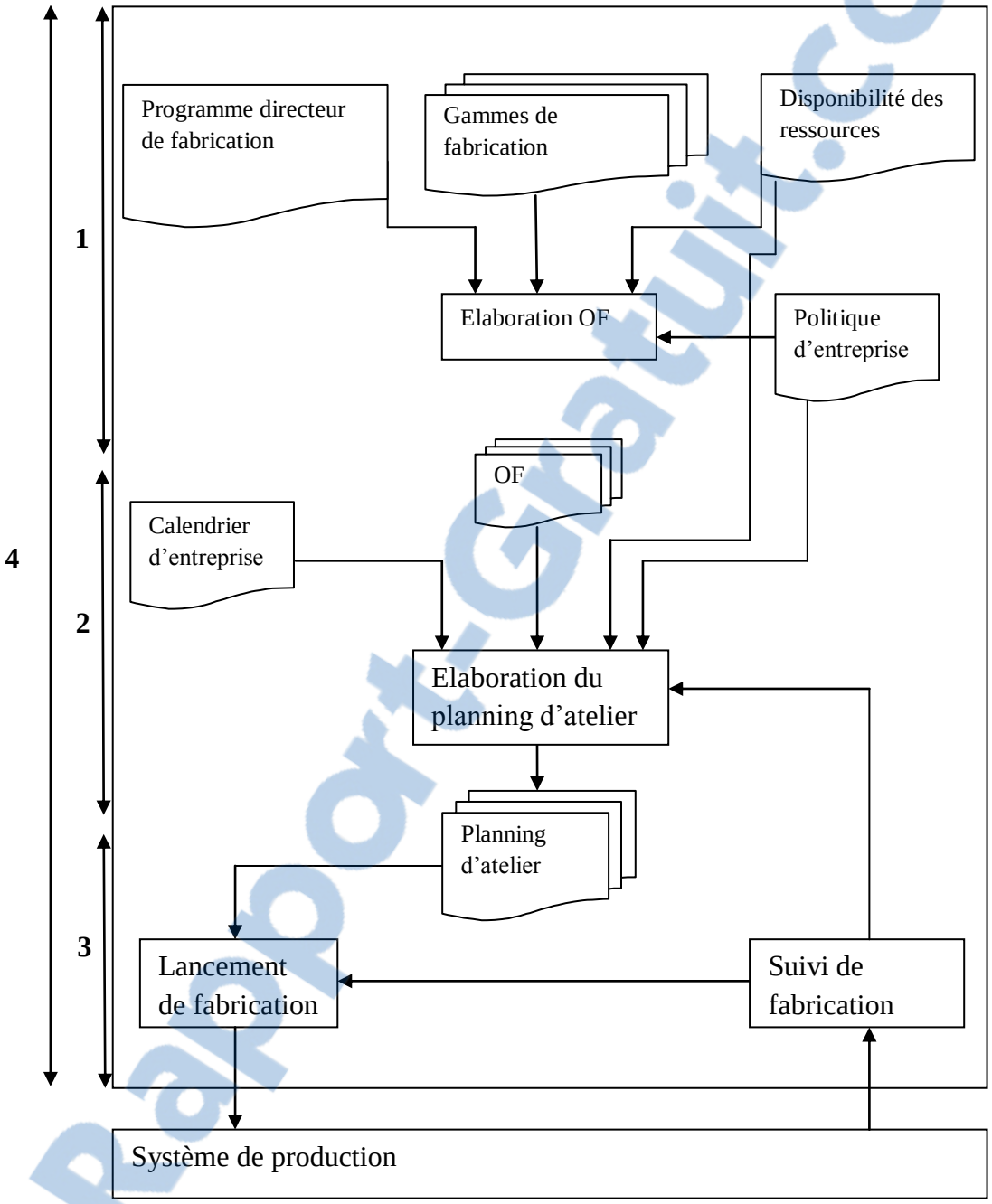
Dans la mesure où des événements aléatoires tels que des pannes ou des commandes imprévues, qui peuvent survenir à tout moment, il est nécessaire de recalculer fréquemment l'ordonnancement, il est difficile de résoudre le problème d'ordonnancement au niveau supérieur (stratégique). Donc, il est résolu au niveau inférieur (opérationnel) (Blondel 2004)

La place de l'ordonnancement varie entre le niveau tactique et le niveau opérationnel (Herrmann 2006). Il s'occupe de la réalisation des décisions venant de niveau supérieur. Son rôle consiste à transformer les décisions de fabrication définies par le programme directeur en instructions d'exécution destinées à contrôler et piloter à court terme l'activité des postes de travail. En sortie de la fonction d'ordonnancement, nous obtienons un planning ou ordonnancement, qui restitue l'affectation des tâches fournies en entrée à des dates précises pour des durées déterminées sur les différentes ressources. Ce planning cherche à satisfaire des objectifs, en respectant le plus possible les contraintes imposées (Javel 2004).

La fonction ordonnancement se décompose en trois sous fonctions **Figure 0.2** (Javel 2004) :

- L'élaboration des ordres de fabrication (OF) : consiste à transformer les informations du programme directeur de production (suggestion de fabrication) en OF.

- L'élaboration du planning d'atelier : consiste à déterminer, en fonction des ordres de fabrication (OF) et de la disponibilité des ressources, le calendrier prévisionnel de fabrication.
- Le lancement et le suivi des opérations de fabrication.



1 : Elaboration des ordres de fabrication
2 : Elaboration du planning
3 : Exécution du planning
4 : Fonction d'ordonnancement

Figure 0.2 Gestion de production (Javel 2004)

4. LES PROBLEMES D'ORDONNANCEMENT

Dans les ateliers de production, un problème d'ordonnancement consiste à attribuer plusieurs tâches à des moyens de production dans le but de réaliser des produits (*Jobs*) tout en respectant les contraintes de production et en optimisant certains critères.

4.1 Définition de l'ordonnancement

En général, la définition d'un problème d'ordonnancement dans un contexte précis nécessite la détermination des caractéristiques particulières des tâches, des ressources, la manière centralisée ou décentralisée de la prise de décision, ainsi que l'ensemble de critères considérés à satisfaire. Nous trouvons dans la littérature plusieurs définitions, parmi lesquelles :

« *Ordonnancer, c'est programmer l'exécution d'une réalisation en attribuant des ressources aux tâches et en fixant leurs dates d'exécution* » (F. Carlier 1997)

(Pinedo 1998) définissent l'ordonnancement comme étant « *l'allocation de ressources (humaines et techniques) aux jobs, tout en fixant les périodes et leurs buts pour optimiser un ou plusieurs objectifs* ».

« *Résoudre un problème d'ordonnancement consiste à ordonner i.e. programmer ou planifier dans le temps, l'exécution des tâches en leurs attribuant les ressources nécessaires, matérielles ou humaines de manière à satisfaire un ou plusieurs critères préalablement définis, tout en respectant les contraintes de réalisation* » (Esquirol et Lopez 1999).

« … *étant donné un ensemble de tâches à accomplir, le problème d'ordonnancement consiste à déterminer quelles opérations doivent être exécutées, et à assigner des dates et des ressources à ces opérations de façon à ce que les tâches soient, dans la mesure du possible, accomplies en temps utile, au moindre coût et dans les meilleures conditions* » (Vacher 2000).

Donc, nous pouvons définir l'ordonnancement comme l'organisation d'exécution des opérations, en assignant une ressource à chaque opération, tout en respectant les contraintes imposées afin d'atteindre un ou plusieurs objectifs.

4.2 Ordonnancement des ateliers

L'ordonnancement des ateliers se base sur l'organisation temporelle du fonctionnement d'atelier, dans le but d'optimiser l'utilisation des ressources humaines et matérielles disponibles, et de respecter les délais exigés.

En général, un problème d'ordonnancement englobe les éléments suivants : les tâches (opérations), les ressources, les contraintes et les critères :

4.2.1 Tâches

Une tâche ou opération (task en anglais) qu'on note « i » est une entité élémentaire de travail localisée dans le temps par une date de début ti et une date de fin c_i, dont la réalisation nécessite une durée P_i telle que $P_i = c_i - t_i$, et qui utilise des ressources k avec une intensité a_{ik} (Esquirol et Lopez 1999).

Généralement, trois paramètres caractérisent une tâche (T'Kindt et Billaut 2006) **Figure 0.3**:

- La durée opératoire P_i(*processing time*) : c'est la durée d'exécution de la tâche.
- La date de disponibilité r_i(*release date*) : c'est la date de début au plus tôt.
- La date d'échéance d_i (*due date*) : c'est la date de fin au plus tard.

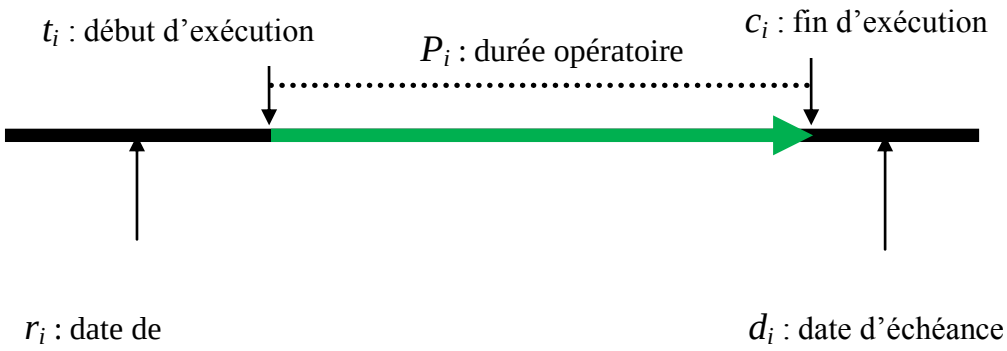


Figure 0.3 Caractéristiques d'une tâche i

Dans le cas où les tâches à exécuter sont données à priori, c'est-à-dire les données de ressources et des opérations (les ressources disponibles, les durées opératoires, les ordres de production, etc.) sont connus à l'avance, le problème d'ordonnancement est dit *statique*. Dans le cas contraire, si l'ensemble des tâches évolue dans le temps (les tâches sont ordonnancées au fur et à mesure qu'elles arrivent), le problème est dit *dynamique*.

Nous distinguons deux types de tâches :

- Les tâches préemptives dont l'exécution peut être divisée en plusieurs intervalles temporels.

- Les tâches indivisibles qui sont exécutées en une seule fois et ne peuvent pas être interrompues avant qu'elles ne soient complètement achevées.

Nous considérons dans notre étude le mode non préemptif, c'est-à-dire, une fois qu'une opération est lancée, son exécution ne peut pas être interrompue.

4.2.2 Gammes

Une gamme opératoire représente une séquence d'opérations ou tâches, afin de déterminer l'ordre de passage des opérations permettant la réalisation d'un produit (il s'agit d'un Job dans notre cas) à travers un ensemble de machines.

Selon l'ordre d'exécution des opérations, Nous distinguons trois types de gammes opératoires :

- La gamme libre : l'ordre est totalement libre,
- La gamme linéaire : l'ordre est entièrement prédéfini,
- La gamme mixte : l'ordre est partiellement déterminé.

Dans notre cas, les gammes opératoires des jobs sont linéaires.

4.2.3 Ressources

Une ressource est un moyen requis humain ou technique utilisé pour exécuter une opération.

Nous trouvons dans les ateliers de production, plusieurs types de ressources :

- Les ressources consommables, qui sont épuisées, une fois qu'elles sont attribuées aux opérations (par exemple, matière première).
- Les ressources renouvelables, qui restent toujours disponibles même après avoir été utilisées par plusieurs opérations (par exemple, machine).
- Les ressources partageables, qui peuvent être partagées entre plusieurs opérations.

Dans notre problème d'ordonnancement des ateliers job shop flexible JSF, nous nous intéressons aux ressources renouvelables, sachant que les ressources représentent des machines.

Nous trouvons une autre classification qui se base sur la façon d'utiliser les ressources :

- Les ressources disjonctives (non partageables), c'est-à-dire les ressources qui ne peuvent exécuter qu'une opération à la fois.
- Les ressources cumulatives (partageables), qui peuvent exécuter plusieurs opérations simultanément (Esquirol et Lopez 1999).

Dans notre cas, les ressources qui représentent des machines sont renouvelables et disjonctives.

4.2.4 Contraintes

Les contraintes représentent les conditions nécessaires pour qu'un ordonnancement soit réalisable. Plus les contraintes sont nombreuses et précises, plus le problème d'ordonnancement devient difficile.

Nous trouvons dans la littérature plusieurs classifications de contraintes, parmi elles :

- Les contraintes de capacité relatives aux conflits sur l'utilisation des ressources, elles sont divisées en deux types :
  - Les contraintes disjonctives où une seule ressource est partagée par une ou plusieurs tâches,
  - Les contraintes cumulatives où plusieurs ressources sont partagées par plusieurs tâches.
- Les contraintes de précédence ou de gamme, précisant l'ordre d'exécution des tâches selon la gamme opératoire,
- Les contraintes liées aux produits finis, telles que : dates de livraison des commandes, dates d'expiration, etc. (Gargouri 2003).

Il existe une autre classification par rapport au système de production :

- **Les contraintes endogènes :** elles sont liées directement au système de production et à ses performances telles que :
  - Les capacités des machines et des moyens de transport,
  - Les dates de disponibilité des machines et des moyens de transport,
  - Les séquences des opérations à exécuter (les gammes opératoires).

- **Les contraintes exogènes :** elles sont indépendantes du système de production, car elles sont imposées extérieurement :
  - les retards possibles accordés pour certains clients,
  - les dates de fin de production au plus tard du produit imposées généralement par des commandes,
  - les priorités de quelques commandes et de quelques clients.

4.2.5 Critères

Un critère est un moyen pour évaluer la qualité de l'ordonnancement établi, ce critère correspond à des exigences qualitatives et quantitatives à satisfaire. Il existe plusieurs genres de critères, mais ils ne sont pas indépendants, certains sont équivalents dans le cas où la solution est optimale pour les deux critères, par exemple : la moyenne des dates de fins des Jobs est équivalente au retard algébrique. (F. Carlier 1997).

- Les critères usuels :
  - La durée totale de l'ordonnancement (le *makespan* ou *Cmax*),
  - Le respect des dates au plus tard (les dates exactes d'achèvement des Job *dj*),
  - La minimisation d'un coût, etc.

- Les critères réguliers :
  - La minimisation de la date de fin effective de la dernière opération de la gamme,
  - La minimisation des dates effectives de fin de toutes les opérations,
  - La minimisation du plus grand retard.

- Les critères irréguliers :
  - L'équilibrage de charge des ressources,
  - L'optimisation des changements d'outils, etc. (Gargouri 2003) (Tangour 2007).

Il existe une relation d'équivalence entre les critères. Par définition, deux critères sont équivalents si une solution optimale pour un est aussi optimale pour l'autre et inversement. Par exemple, la moyenne des dates de fins est équivalente au retard algébrique moyen (F. Carlier 1997).

4.2.6 Les relations entre les critères

Les premières études ont traité les problèmes d'ordonnancement en utilisant un seul critère d'optimisation, mais avec l'évolution des environnements manufacturiers, les objectifs des entreprises se sont diversifiés et le processus d'ordonnancement est devenu de plus en plus multicritères (Letouzey 2001).

Mais il existe plusieurs relations entre ces critères, telles que la relation d'équivalence, de réduction (lorsqu'un critère peut être déduit d'autres critères), par exemple un problème avec le critère *Lmax* (retard algébrique) peut se réduire en $\sum_{i=1}^{N} T_i$ (la somme des retards absolus de l'ensemble des jobs).

5. CLASSIFICATION DES ATELIERS

Classifier les problèmes d'ordonnancement revient à étudier les différents types de machines utilisées ainsi que la manière avec laquelle l'atelier est organisé.

Les ateliers dans les systèmes de production sont nombreux, ils se diffèrent par le nombre de machines, la nature des jobs (par exemple la composition des jobs d'une ou de plusieurs tâches), l'unicité ou la diversité du routage des jobs au niveau des machines, ou par la flexibilité des ressources (c'est-à-dire la possibilité d'avoir un sous-ensemble de machines candidates dans lesquelles une telle tâche ou opération peut être traitée), dans le cas d'une flexibilité partielle, sinon n'importe quelle tâche peut être exécutée sur n'importe quelle ressource. Les ressources dans les ateliers de production peuvent dotés de différentes organisations :

- Les organisations à ressources unique : C'est l'organisation la plus simple, or elle représente un domaine de recherche pour les systèmes informatique dont la ressource partagée est souvent nommée : ressource critique, comme par exemple, un seul processeur, un disque dure, imprimante, etc.
- Les organisations à multi-étages parallèles : il existe au niveau de chaque étage, des machines capables d'exécuter les mêmes opérations. Ces machines peuvent être identiques, uniformes ou différentes. Les étages sont disposés parallèlement, ce qui

permet d'avoir plusieurs cheminements possibles en même temps. La **Figure 0.4** (T'Kindt et Billaut 2006) montre un schéma général d'une organisation multi-étages avec trois étages parallèles.

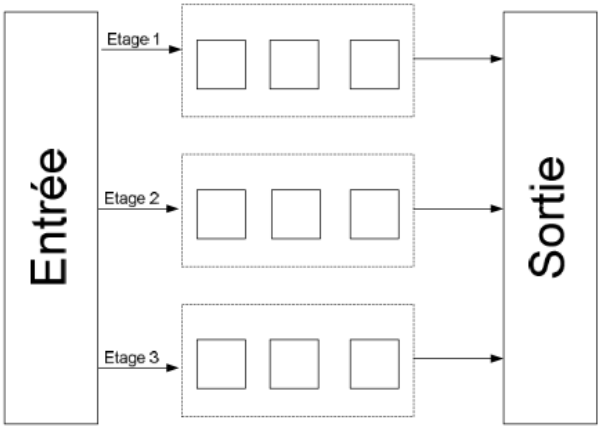


Figure 0.4 Organisation multi-étage parallèles (T'Kindt et Billaut 2006)

- Les organisations à ressources multiples : cette organisation est dotée de plusieurs ressources exécutants plusieurs taches. Le type de cheminement des taches donne trois sous organisations appelées Flow Shop, Job Shop et Open Shop :

Ateliers Flow-Shop

Dans ce type d'atelier, la ligne de fabrication est constituée de plusieurs machines en série, de telle sorte que toutes les opérations de tous les jobs passent par toutes les machines en respectant le même ordre (**Figure 0.5**) (Duvivier 2000). C'est pour cela que ces ateliers sont appelés les ateliers à cheminement unique. Si il existe plusieurs exemplaires identiques et parallèles de la même machine, l'atelier devient Flow-Shop Hybride (**Figure 0.6**) (Duvivier 2000).

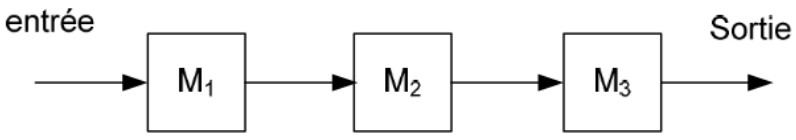


Figure 0.5 Flow Shop à trois machines (Duvivier 2000)

Pour un atelier Flow Shop avec composé de deux machines, il existe des méthodes exactes et efficaces (dans un temps polynomial). Au delà, les méthodes approchées sont utiles telles que les méta-heuristiques pour un résultat de bonne qualité.

La **Figure 0.6** montre un Flow Shop Hybride avec 3 étages. Chaque job doit être traité sur une seule machine à chaque étage et l'ordre de passage d'un étage à un autre est le même (étage 1, étage 2,…, étage m). Un autre problème d'affectation des jobs aux machines s'ajoute au problème d'ordonnancement des jobs.

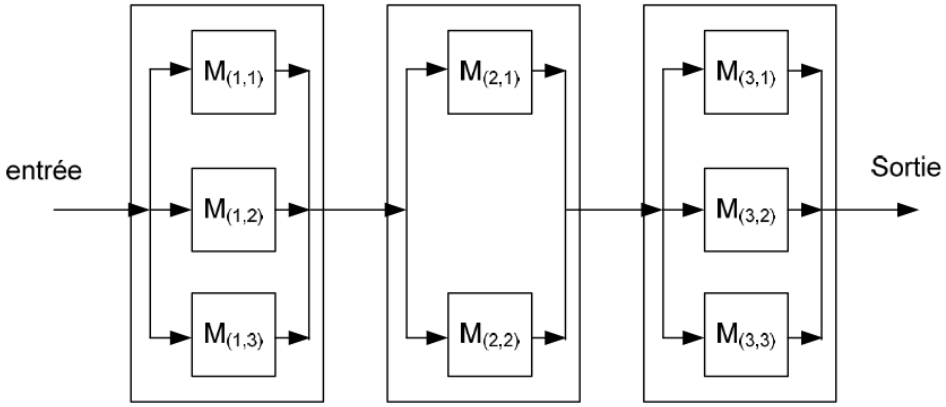


Figure 0.6 Flow Shop hybride constitué de trois étages (Duvivier 2000)

Ateliers Job Shop et job shop flexible

Contrairement au type d'atelier précédent, l'atelier Job Shop se caractérise par un cheminement multiple, puisque les opérations de chaque Job peuvent emprunter divers chemins (routage des opérations).

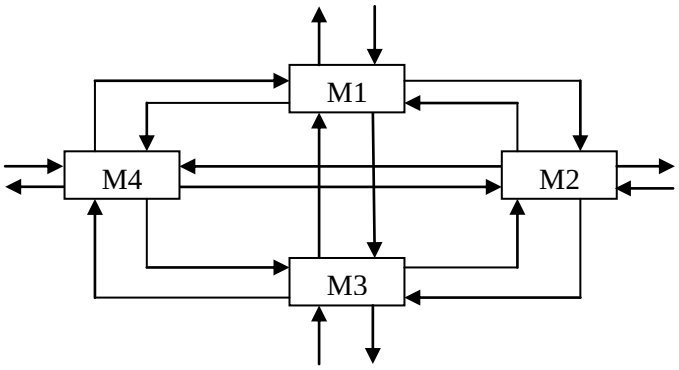
Généralement, il existe deux organisations d'atelier Job Shop **Figure 0.7** (Duvivier 2000) :

1. Organisation simple : s'il existe pour chaque machine un seul exemplaire,
2. Organisation hybride : s'il existe au moins une machine disposant de plusieurs exemplaires.

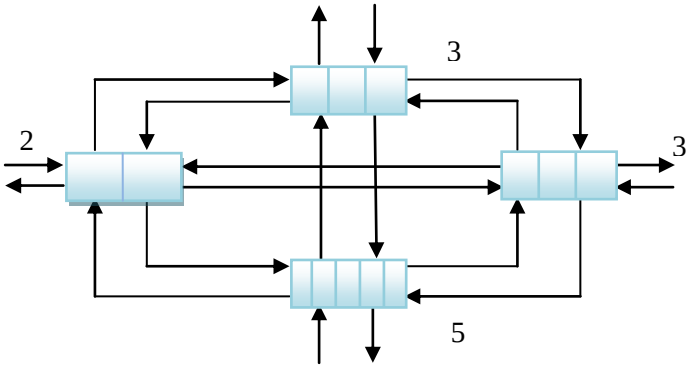
L'atelier Job Shop Flexible est une extension du problème classique décrit précédemment. La flexibilité est due aux ressources, c'est-à-dire, l'attribution d'un sous-ensemble de ressources (machines candidates) pour le traitement de chaque opération de telle sorte que le temps opératoire de chaque opération dépend de la machine candidate sélectionnée.

En effet, il existe plusieurs degrés de flexibilité : dans la flexibilité faible, quelques opérations sont traitables dans quelques machines appelées machines candidates. Ensuite, dans la flexibilité moyenne et forte le nombre d'opérations traitables dans plusieurs machines

devient de plus en plus important. En Arrivant à la flexibilité extrême (totale), n'importe quelle opération est traitable sur n'importe quelle machine.



(A) Job Shop simple avec 4 machines



(B) Job Shop hybride à 4 postes de travail

Figure 0.7 Job Shop simple et hybride (Duvivier 2000)

Ateliers Open Shop

Dans ce type d'ateliers, l'acheminement d'opérations est multiple et libre, autrement dit, il n'existe aucun ordre d'exécution des opérations (les gammes sont libres). Nous trouvons ce type d'atelier dans le cas où la fabrication de chaque produit se traduit par le traitement de plusieurs opérations, dont l'ordre est totalement libre.

Un autre problème d'ordonnancement existe, nommé : RCPSP (Resource Constrained Project Scheduling Problem). Les ressources dans ce problème sont disponibles en nombre limité et typées (nous ne pouvons pas utiliser une ressource à la place de l'autre). Les ressources renouvelables sont disponibles pour les opérations restantes et les ressources non renouvelables qui ne peuvent être utilisées qu'une seule fois.

La **Figure 0.8** (Bertel 2001) montre les relations existantes entre les différentes organisations et donne une vue générale de la typologie des problèmes d'optimisation.

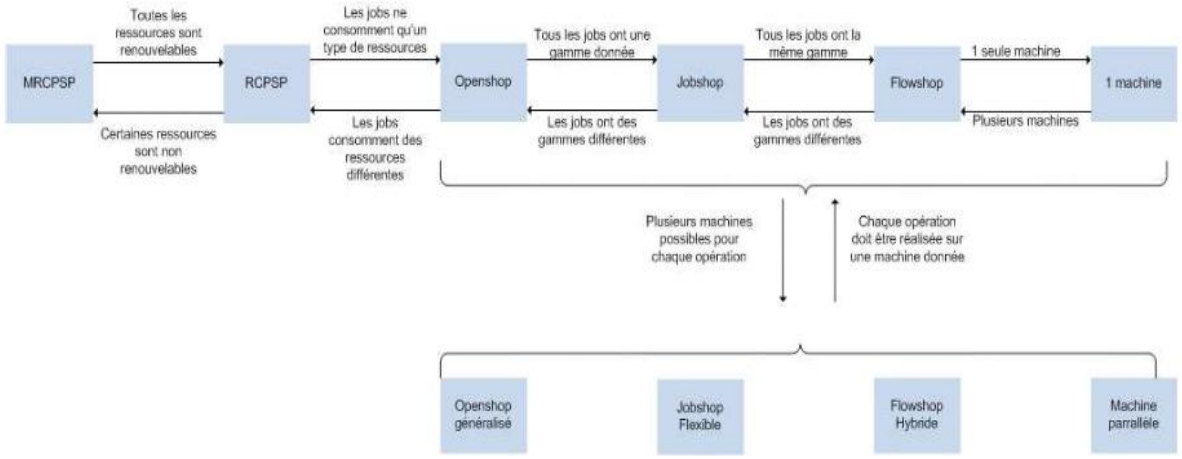


Figure 0.8 typologie des problèmes d'ordonnancement (Bertel 2001)

6. COMPLEXITE

Dans les problèmes d'ordonnancement, la complexité est liée à la complexité des méthodes de résolution et des algorithmes utilisés (Garey et Johnson 1979) (J. Carlier 1984) (F. Carlier 1997), (Charon 1996). La complexité de quelques types de problèmes d'ordonnancement dotés d'une taille relativement grande devient importante. (Tangour 2007).

La complexité d'ordonnancement peut être divisée en deux grandes catégories : algorithmique et problématique (Tangour 2007).

6.1 La complexité algorithmique

L'objectif de la théorie de complexité est d'analyser les coûts de résolutions surtout en termes de temps de calcul. Elle vise aussi à classifier les problèmes en plusieurs niveaux de difficulté. Une étude a prouvé que les problèmes d'ordonnancement sont des problèmes difficiles (Letouzey 2001).

« La complexité d'un algorithme A est une fonction $C_A N$, donnant le nombre d'instructions caractéristiques exécutées par A dans le pire des cas, pour une donnée de taille N » (Lacomme, Prins et Sevaux 2003).

En général, la complexité algorithmique se mesure par rapport à deux paramètres :

– Le temps alloué pour l'exécution de l'algorithme : il est relatif au nombre d'instructions à exécuter ainsi qu'à la taille des données manipulées.

– L'espace mémoire requis : associé à la taille d'instance d'un problème donné. (Paschos 2005).

6.2 La complexité problématique

La complexité problématique est relative au problème à résoudre ainsi que la méthode de résolution adoptée pour élaborer la solution optimale par rapport aux critères retenus.

– Un problème de décision comprend deux parties : la partie relative aux données du problème et un processus binaire ayant « oui » ou « non » comme réponse possible.
– Un problème de recherche est un problème constitué d'un ensemble de données dont chacun représente un ensemble de solutions. Donc, la résolution d'un problème de recherche consiste à trouver pour chaque ensemble de données D des solutions S associées. Un problème d'optimisation est un problème de recherche en associant à chaque solution une valeur qualitative. À chaque problème d'optimisation, nous pouvons associer un problème de décision (par exemple l'exclusion ou l'inclusion d'une solution dans les futures générations pour les AG), donc l'étude de la complexité du problème de décision peut donner des indications au problème d'optimisation associé. (Charon 1996).

La théorie de complexité permet de classer les problèmes en deux classes *P* et *NP*. La classe *P* regroupe les problèmes qui peuvent être résolus par des algorithmes polynomiaux. Un algorithme est dit polynomial, lorsque son temps d'exécution est borné par $O(P(x))$ où *P* est un polynôme et *x* est la longueur d'entrée d'une instance du problème. Les algorithmes polynomiaux représentent une classe stable et la composition de deux algorithmes polynomiaux engendre un algorithme polynomial, et un algorithme construit polynomialement à partir d'appels à des procédures de complexité polynomiale, reste polynomial. (Lacomme, Prins et Sevaux 2003).

Les algorithmes dont la complexité ne peut pas être bornée polynomialement sont qualifiés d'exponentiels et correspondent à la classe *NP* '***N**on determinist **P**olynomial time*'. Les problèmes dans NP peuvent être résolus en énumérant l'ensemble des solutions possibles et en les testant à l'aide d'un algorithme polynomial. Tous les problèmes de décision dont nous pouvons associer à chacun d'eux un ensemble de solutions potentielles (de cardinal au pire exponentiel) de telle sorte qu'il existe un algorithme polynomial qui peut vérifier, si la solution trouvée est valable ou pas.

Un problème de décision est dit *NP-Complet* s'il appartient à la classe *NP* et il est résolu, au mieux, en un temps exponentiel.

Un problème d'optimisation est dit *NP-Difficile*, si le problème de décision associé est *NP-complet*. La **Figure 0.9** montre les différentes classes de complexité.

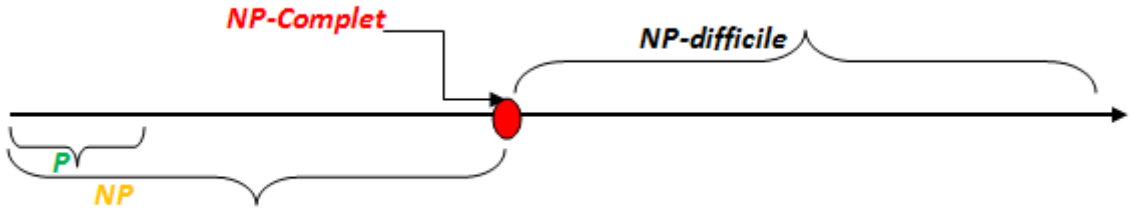


Figure 0.9 Représentation des Classes P, NP, NP-complet et NP-difficile

7. CLASSIFICATION D'ORDONNANCEMENT

Il existe dans la littérature plusieurs classifications des problèmes d'ordonnancement. En effet, chaque ordonnancement est lié à ses propres critères, tels que les incertitudes. Nous présentons dans cette thèse deux classifications des problèmes d'ordonnancement, la première est liée au décalage des opérations (avancement d'une opération dans le temps) et son impact sur les autres opérations. La deuxième classification est basée sur les incertitudes relatives aux ressources.

7.1 Classification des problèmes d'ordonnancement

Un des paramètres primordiaux pour la génération d'un ordonnancement optimale dans les problèmes d'ordonnancement Job Shop, est l'exploitation maximale du temps, car un simple décalage des opérations retardées peut éviter les trous de temps « *temps morts* » afin de préparer un ordonnancement « compact ». En effet, Plus un ordonnancement est compact, meilleure est sa qualité (Duvivier 2000).

Cette notion de compacité qui est un objectif majeur de tout ordonnancement est la base de distinction de plusieurs classes d'ordonnancement : admissible, semi-actif, actif, sans délai (Pinedo 1998) (T'Kindt et Billaut 2006).

7.1.1 Ordonnancement admissible

Si toutes les contraintes du problème sont bien respectées, l'ordonnancement est dit : « *Admissible* ».

Dans certains cas, des décalages à gauche sur certaines opérations sont nécessaires. Selon que l'ordre des opérations reste inchangé ou non, nous distinguons deux cas (Pinedo 1998) :

- Décalage à gauche *local* : l'avancement du début d'une opération ne remet pas en cause l'ordre des autres opérations,

- Décalage à gauche *global* : l'avancement du début d'une opération engendre une modification au niveau de l'ordre relatif aux deux opérations au minimum.

7.1.2 Ordonnancement semi-actif

Si aucun décalage local n'est possible, l'ordonnancement est dit *Semi-actif*. Donc, aucune opération ne peut être exécutée en plus tôt sans modifier l'ordre relatif au moins de deux opérations.

7.1.3 Ordonnancement actif

Si aucun décalage à gauche que ce soit *local* ou *global* n'est possible, l'ordonnancement est dit *Actif*. En conséquence, il est impossible d'avancer une opération sans reporter le début d'une autre opération (Pinedo 1998).

7.1.4 Ordonnancement sans délais

Un ordonnancement est dit 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 (T'Kindt et Billaut 2006). À noter que la transformation en un ordonnancement sans délai peut mener à une solution plus mauvaise du point de vue makespan (Esquirol et Lopez 1999).

Il existe une relation d'inclusion entre les différentes classes d'ordonnancement précédentes (**Figure 0.10**) :

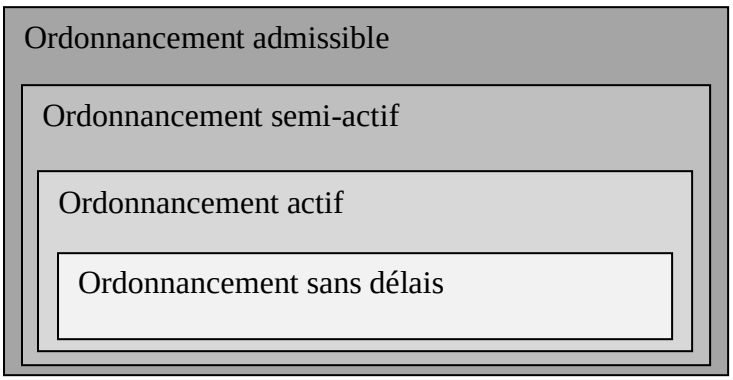


Figure 0.10 Relation d'inclusion entre les différentes classes d'ordonnancement (Pinedo 1998)

D'après la **Figure 0.10**, nous constatons que les ordonnancements semi-actif, actif et sans délais sont inclus dans la classe d'ordonnancement admissible, puisqu'ils vérifient toutes les contraintes d'un ordonnancement admissible.

7.2 L'ordonnancement sous l'incertitude

Récemment, avec la dynamique accrue des environnements des systèmes de production, l'ampleur des facteurs de risques est de plus en plus persistante, car d'une part, l'absence de la connaissance requise pour décrire un système engendre des incomplétudes qui peuvent

conduire à des incertitudes et d'autre part, les imprécisions qui sont causées par la difficulté d'affirmer la validité d'une telle connaissance.

Toutes ces raisons montrent l'importance de la prise en considération des incertitudes et des imprécisions dans l'élaboration d'ordonnancement au niveau des systèmes de production.

En général, les approches d'ordonnancement sont divisées en trois catégories (Chaari 2010):

- Les approches Proactives : où l'ordonnancement se fait avant l'exécution du plan, c'est-à-dire au niveau de la phase « Offline ».
- Les approches Réactives : elles sont adaptées aux environnements dynamiques pour lesquels les décisions doivent être prises en temps réel.
- Les approches Hybrides : Ces approches tentent à la fois d'anticiper les risques via un ordonnancement déterministe, et de s'adapter avec la dynamique des environnements en essayant de réordonnancer en temps réel le plan initialement préparé. Ce type d'approche est subdivisé en deux sous-approches : « Approches Prédictives-Réactives » et « Approches Proactives-Réactives.

8. MODELISATION ET REPRESENTATION DES PROBLEMES D'ORDONNANCEMENT

8.1 Notations

Il existe dans la littérature plusieurs méthodes de notation des problèmes d'ordonnancement. Par exemple la méthode de (Garey et Johnson 1979) (Peter Brucker 1994) dans laquelle un problème d'ordonnancement est présenté de la manière suivante :

Trois champs sont utilisés : (a | b | c)

– Le champ « a » pour décrire le type d'atelier, le nombre de job à réaliser et le nombre de machines disponibles dans l'atelier.

– Le champ « b » pour définir les contraintes du problème et les différentes hypothèses sur le mode d'exécution des tâches (précédence, préemption, etc.).

– Le champ « c » pour préciser le ou les critères à optimiser.

Voici un exemple d'une notation ;
(J, 9, 5 | $Prec$, r_i | C_{max})
Dans cet exemple, le type d'atelier est Job Shop J, composé de 9 jobs à cinq machines. Le deuxième champ montre que les jobs ont une contrainte de précédence, *Prec* et une contrainte r_i de date de début au plus tôt. Une autre contrainte s'ajoute relative à la

préemption qui est interdite (Prem n'est pas mentionné dans le champ *prem*). La fonction objective qui est la minimisation du makespan C_{max} est précisée par le troisième champ.

8.2 Modélisation

En général, afin de bien présenter la réalité d'un problème, la modélisation intervient pour fournir une écriture simplifiée de toutes les données relatives au problème considéré.

Il existe dans la littérature, deux méthodes de modélisation : les méthodes graphiques et les méthodes mathématiques :

8.2.1 La modélisation graphique

Avec l'apparition de plusieurs techniques de modélisation graphique, surtout les *Réseaux De Pétri 'RDP'* (Chrétienne 1983), la modélisation graphique est devenue plus précise, permettant de traduire plusieurs notions fondamentales relatives au problème d'ordonnancement considéré.

La modélisation graphique peut se faire par le *graphe potentiel-tâches*, ce dernier est constitué des éléments suivants :

- Des nœuds représentant les tâches à réaliser,
- Des arcs conjonctifs pour la précision des contraintes de précédence (en indiquant la durée de tâches),
- Des arcs disjonctifs indiquant les contraintes de ressources (Roy 1969), (Gotha 1993), (Watson, et al. 2003).

L'apparition des réseaux de pétri a permit de traduire plusieurs notions fondamentales liées à l'ordonnancement telles que (Tangour 2007):

- Les conflits sur les ressources,
- Les durées opératoires certaines (RDP temporisé),
- Les durées opératoires aléatoires (RDP stochastique),
- les gammes opératoires,
- Les disponibilités, les multiplicités et les capacités des ressources (RDP synchronisé et à capacité),
- La répétitive (RDP cyclique).

Exemple :

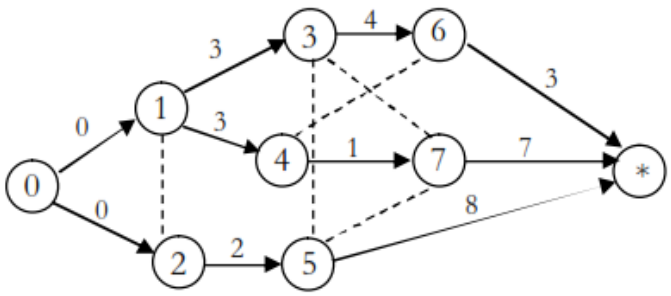


Figure 0.11 Graph potentiel – tâches (Kacem, Hammadi et Borne 2002)

8.2.2 La modélisation mathématique

Les modèles mathématiques visent à trouver des équations mathématiques pour décrire les données, les contraintes ainsi que les critères utilisés pour l'optimisation des solutions. L'avantage de ce type de modélisation est d'obtenir des expressions mathématiques simples et robustes, qui peuvent être exploitées facilement en programmation.
Par exemple (Tangour 2007), Les données suivantes sont relatives à l'exemple présenté par la **Table 0.1**, $i \in \{1,2,3,4,5,6,7\}$, sont l'ensemble des tâches. L'objectif consiste à :

- Calculer t_i (date de début d'exécution de la tâche i).
- Minimiser la date de fin de la dernière tâche, le C_{max} .
- Respecter les contraintes présentées dans le tableau 1.1, sachant que la durée opératoire d'une tâche i est représentée par P_i :

Contraintes de données	**Contraintes de précédence**	**Contraintes de ressources**
$P_1 = 3$	$t_1 + p_1 \leq t_3$	$t_1 + p_1 \leq t_2$ ou $t_2 + p_2 \leq t_1$
$P_2 = 2$		
$P_3 = 4$	$t_1 + p_1 \leq t_4$	$t_3 + p_3 \leq t_5$ ou $t_5 + p_5 \leq t_3$
$P_4 = 1$		$t_3 + p_3 \leq t_7$ ou $t_7 + p_7 \leq t_3$
$P_5 = 8$	$t_3 + p_3 \leq t_6$	$t_7 + p_7 \leq t_5$ ou $t_5 + p_5 \leq t_7$
$P_6 = 3$	$t_4 + p_4 \leq t_7$	
$P_7 = 7$	$t_2 + p_2 \leq t_5$	$t_4 + p_4 \leq t_6$ ou $t_6 + p_6 \leq t_4$

Table 0.1 Données et contraintes de l'exemple de la **Figure 0.11**

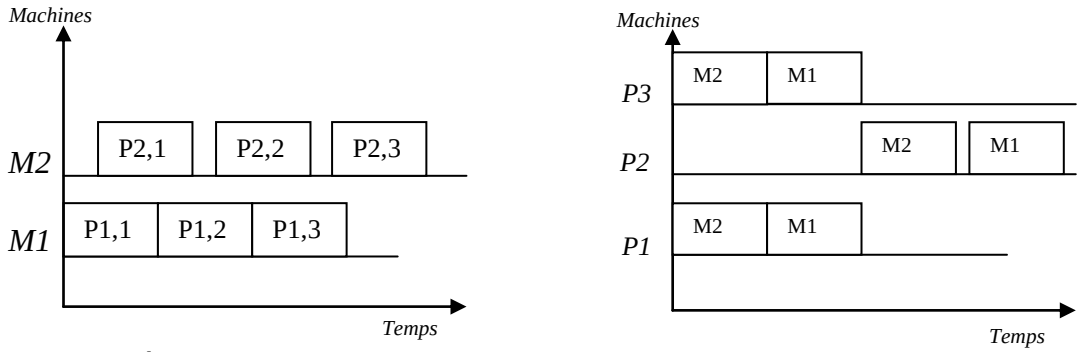
8.3 Représentation des solutions

Il existe plusieurs techniques de représentation des solutions, mais la plus souvent utilisée, par rapport à sa simplicité, est la représentation par le diagramme de *Gantt*.

Le diagramme de *Gantt* a deux méthodes de représentation :

- La représentation par produit,
- La représentation par machine.

Exemple : J 2,2 |Précé | Cmax : problème Job Shop avec deux jobs et deux opérations pour chaque job.



Représentation par machines

Représentation par produits

Figure 0.12 Diagrammes de Gantt

9. PARTICULARITES D'ORDONNANCEMENT DANS LES ATELIERS JOB SHOP FLEXIBLES

Le problème Job Shop Flexible (JSF) est une extension du problème classique d'ordonnancement Job Shop. Ce type de problème représente un modèle d'ateliers largement utilisé dans les systèmes manufacturiers réels. Le problème JSF comprend plusieurs Jobs (*Pièces* à produire) et plusieurs machines. Chaque Job est composé d'un ensemble d'opérations (*Gamme opératoire d'un Job*).

Le problème JSF est *flexible*, car il permet de traiter chaque opération sur n'importe quelle machine incluse dans un sous-ensemble de machines candidates. Il existe plusieurs degrés de flexibilité de ressources (machines) :

- la flexibilité partielle : si chaque opération peut être exécutée sur quelques machines candidates seulement, parmi l'ensemble de machines,

- la flexibilité totale : si chaque opération peut être exécutée sur n'importe quelle machine (l'ensemble de machines candidates égal à l'ensemble global de machines).

D'autres problèmes peuvent être considérés comme fortement flexibles, si chaque opération peut être exécutée sur la plupart des machines présentes dans l'atelier.

Deux cas principaux sont traités dans la littérature :

- Les durées opératoires sont identiques,
- Les durées opératoires dépendent des machines candidates.

Dans notre cas, les durées opératoires dépendent des machines candidates.

En plus du problème de séquencement, la flexibilité du problème JSF engendre un autre problème d'assignement des opérations aux machines candidates, donc le problème d'ordonnancement peut être résumé en deux sous-problèmes :

- Problème d'assignement : consiste à affecter une machine à chaque opération,
- Problème de séquencement : consiste à établir un ordre d'exécution d'opérations au niveau de chaque machine.

La résolution du problème d'ordonnancement JSF peut se faire de deux manières :
- L'approche *intégrée* : dans cette approche, les deux sous-problèmes d'assignement et de séquencement sont résolus simultanément.
- L'approche *hiérarchique* : la résolution se fait par phases : tout d'abord résoudre le premier sous problème d'assignement, après nous passons à la deuxième phase relative au deuxième sous-problème de séquencement d'opérations.

L'approche que nous avons adoptée est intégrée, elle permet de résoudre les deux sous problèmes d'assignement et de séquencement simultanément.

10. CONCLUSION

Quelque soit le type de système de production, la préparation d'un bon ordonnancement est un processus très important qui influe directement la productivité du système. En effet, un mauvais ordonnancement signifie une utilisation non adéquate de ressources de production et donc, une consommation physique (ressources) et temporelle (temps de production) élevée, qui se traduit par une performance basse du système de production. Mais, l'ordonnancement peut être perturbé par des événements indésirables, telles que les pannes des machines, les demandes non prévues, etc. Dans ces cas, l'ordonnancement doit être recalculé afin de s'adapter avec ces perturbations.

Le problème d'ordonnancement est un problème combinatoire qui est classé parmi les problèmes d'optimisation les plus difficiles. En effet, la complexité de l'ordonnancement dépend du domaine et de la taille du problème traité. Plusieurs approches de résolution du

problème d'ordonnancement existent, qui sont classées en deux catégories : approches exactes et approches approximatives. Le chapitre suivant illustrera ces approches.

CHAPITRE 2 : PROBLEMES D'OPTIMISATION ET LEURS METHODES DE RESOLUTION

1. INTRODUCTION

L'objectif d'un problème d'optimisation est de sélectionner la ou les meilleures solutions qui répondent aux contraintes du problème ainsi qu'aux objectifs définis au départ. Cette sélection repose sur le principe de comparaison entre les différentes solutions afin d'en tirer les meilleures et d'éliminer les autres.

En général, les problèmes d'ordonnancement sont considérés comme des problèmes d'optimisation, parce que leur objective est de minimiser ou maximiser une fonction objective tout en respectant certain critères.

La notion de l'optimisation multi-objectif remonte à un ouvrage de W. Pareto sur l'économie politique en 1906, dans lequel, la notion de l'optimum multi-objectif a été définie pour la première fois.

Il existe deux approches d'optimisation multi-objectif :

- Résolution vectorielle du problème, dans lequel l'expression du problème n'est pas modifiée.
- Simplification du problème en un problème d'optimisation mono objectif.

Dans ce chapitre, nous présentons les notions de base relatives aux problèmes d'optimisation ensuite, nous passerons aux différentes méthodes de résolutions des problèmes d'optimisation qui sont les méthodes exactes et les méthodes approchées en mettant l'accent sur les méta-heuristiques.

2. NOTIONS RELATIVES AUX PROBLEMES D'OPTIMISATION

2.1 La fonction objectif

C'est une fonction mathématique composée de variables de décision qui représentent le modèle physique du système. Cette fonction coût qu'on cherche à optimiser est notée f

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 f. (Collette et Siarry 2002).

2.3 Les types de minima

Nous distinguons deux types de minima (**Figure 0.1**) : les minimas locaux et les minimas globaux :

– Minimum global :

Un point x* est un minimum global de la fonction f;

Si : $f(x^*) < f(x)$ $\forall$ x Tel que $x \neq x^*$

– Minimum local :

Un point x* est un minimum global de la fonction f;

Si : $f(x^*) < f(x)$ $\forall x \in V(x^*)$ et $x^* \neq x$, où $V(x^*)$ définit un voisinage de x*.

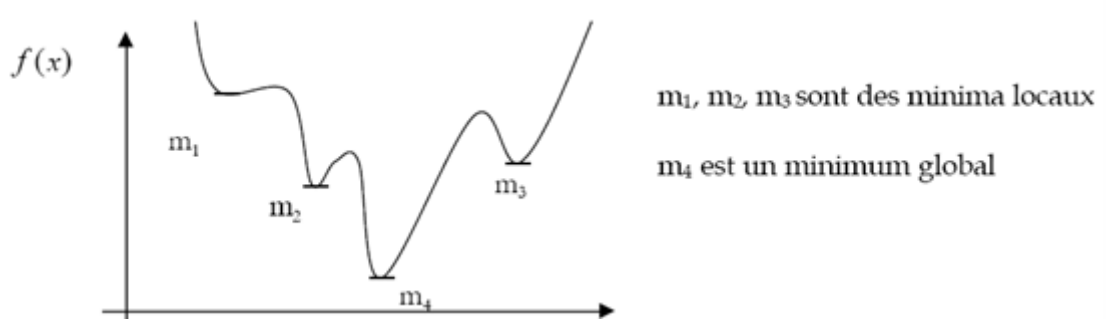


Figure 0.1 Les deux types de minima (Tangour 2007)

Les problèmes d'optimisation peuvent être classés selon :

- Le nombre de variables de décision : problèmes mono variable ou multi variables.
- Le type de variable de décision :
  - Nombre entier : nombre discret
  - Nombre réel continu : problème continu
  - Permutation sur un nombre fini de nombre : combinatoire.

2.4 Les problèmes mono-objectifs et multi-objectifs

Le traitement de l'ordonnancement dans la littérature s'est tout d'abord orienté vers une optimisation monocritère. L'environnement manufacturier évoluant rapidement, les objectifs des entreprises se sont diversifiés et le processus d'ordonnancement est devenu de plus en plus multicritères (Letouzey 2001).

A. Problèmes mono-objectifs

Mathématiquement, un problème d'optimisation mono-objectif est présenté de la manière suivante :

$$\begin{cases} \text{Minimiser } f(\vec{x}) \quad \text{(fonction à optimiser)} & \vec{x} \in \mathcal{R}^n,\ \vec{f}(\vec{x}) \in \mathcal{R} \\ \text{Avec } \vec{g}(\vec{x}) \leq 0 \quad \text{(m contraintes d'inégalités)} & \vec{g}(\vec{x}) \in \mathcal{R}^m \\ \text{Et } \vec{h}(\vec{x}) = 0 \quad \text{(n contraintes d'égalités)} & \vec{h}(\vec{x}) \in \mathcal{R}^n \end{cases}$$

B. Problèmes multi-objectifs

Lorsqu'on modélise un problème, il est souvent nécessaire de satisfaire plusieurs objectifs (Boukef 2009). C'est le cas d'optimisation multi-objectifs.

Beaucoup de problèmes dans le monde réel nécessitent l'optimisation simultanée de plusieurs fonctions objectifs. Ces fonctions sont souvent contradictoires.

D'un point de vue mathématique, un problème d'optimisation multi-objectif, se présente, dans le cas ou le vecteur $\vec{f}$ regroupe k fonctions objectif, de la façon suivante (Collette et Siarry 2002):

$$\begin{cases} \text{Minimiser } \overrightarrow{f}(\vec{x}) & \vec{x} \in \mathcal{R}^m,\ \overrightarrow{f}(\vec{x}) \in \mathcal{R}^k \\ \text{Avec } \overrightarrow{g}(\vec{x}) \leq 0 & \overrightarrow{g}(\vec{x}) \in \mathcal{R}^m \\ \text{Et } \overrightarrow{h}(\vec{x}) = 0 & \overrightarrow{h}(\vec{x}) \in \mathcal{R}^p \end{cases}$$

Quand nous traitons un problème d'optimisation avec plusieurs objectif contradictoires, nous aurons un ensemble de solutions optimales, à la place d'une seule solution. Dans ce cas, il n'existe aucune solution qui est considérée comme meilleure que d'autre, en respectant toutes les fonctions objectif. C'est la raison pour laquelle nous parlons d'une optimalité de plusieurs solutions. Ces solutions optimales sont connues comme Pareto-Optimal solutions.

L'optimisation multi-objectif repose sur le concept de *dominance* par une comparaison entre deux solutions. Si une solution n'est pas dominée par d'autres solutions, cette solution est appelée : *non-dominated solutions*. Dans la sous section suivante, nous expliquons la notion de dominance.

2.5 Notion de dominance

Le concept de dominance est utilisé dans les problèmes d'optimisation multi-objectif. L'importance du concept de dominance d'un sous-ensemble de solutions est de limiter la complexité algorithmique en limitant l'espace de recherche d'une solution optimale au sein d'un grand ensemble de solutions.

Pour trouver une solution optimale dans les problèmes d'optimisation multi-objectif, 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 (Jouglet, Baptiste et Carlier 2002). La plus utilisée est celle définie au « sens de Pareto » (Barichard 2003).

Ainsi, la résolution d'un problème d'optimisation multi-objectif 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 (Collette et Siarry 2002).

Nous disons que le vecteur $\vec{x}_1$ domine le vecteur $\vec{x}_2$ si :

· $\vec{x}_1$ est au moins aussi bon que $\vec{x}_2$ dans tous les objectifs,

· $\vec{x}_1$ est strictement meilleur que $\vec{x}_2$ dans au moins un objectif.

Dominance forte :

Nous disons que $\vec{x}_1$ domine fortement $\vec{x}_2$ si $\vec{x}_1$ est meilleur que $\vec{x}_2$ dans tous les objectifs.

Les solutions qui dominent les autres mais ne se dominent pas entre elles, sont appelées solutions optimales au sens de Pareto (Barichard 2003).

La **Figure 0.2** illustre le concept of Pareto pour deux fonctions objectif à minimiser. L'ensemble des solutions non dominées est appelé : *Pareto Optimal Front*. Ces solutions sont aussi appelées : *non-dominated* ou *non-inferior points*.

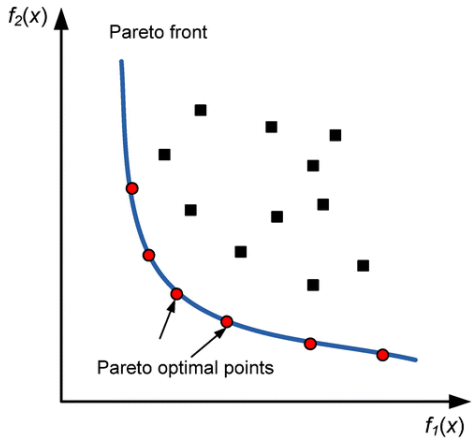


Figure 0.2 Concept d'optimalité Pareto (Barichard 2003)

3. METHODES DE RESOLUTION DES PROBLEMES D'OPTIMISATION

La difficulté de résolution des problèmes d'optimisation a poussé les chercheurs à proposer plusieurs méthodes. Traditionnellement, les problèmes d'optimisation sont résolus à travers les *méthodes exactes*, mais lorsque la taille du problème devient importante, la recherche d'une solution optimale dans un espace important devient un processus difficile à achever dans un temps raisonnable, car l'exploration complète de l'espace de solution prend un temps très important vu le volume de calcul. C'est pour cette raison que les chercheurs ont adopté une autre catégorie de méthodes de résolution, nommée, les méthodes *approximatives* ou *approchées*, ces dernières sont basées sur des techniques d'exploration de l'espace de solution

, qui sont plus rapides que celles utilisées dans les méthodes exactes et qui peuvent fournir des solutions quasi optimales ou optimales dans un temps acceptable.

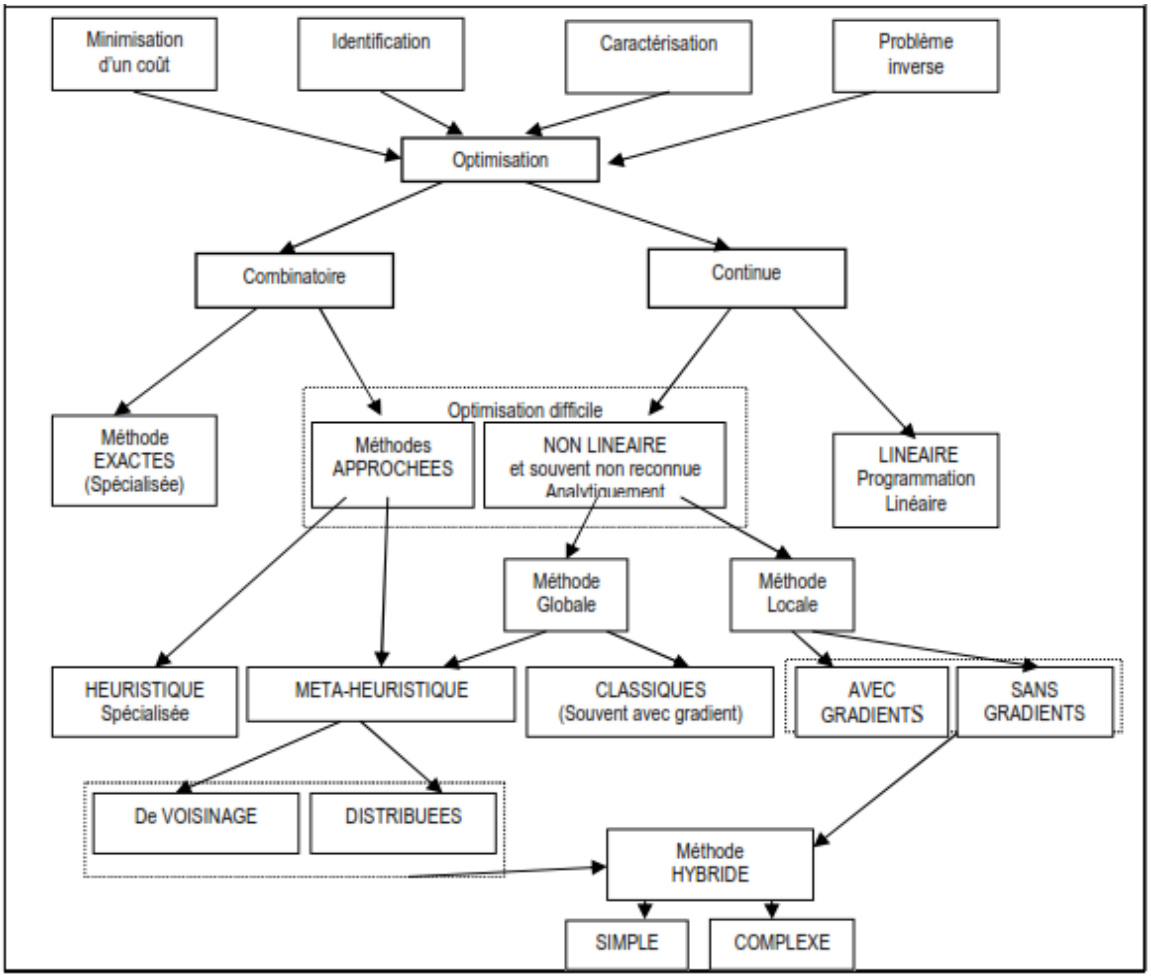


Figure 0.3 une hiérarchie des méthodes d'optimisation (Collette et Siarry 2002)

Le tableau suivant (**Table 0.1**) énumère quelques principales méthodes exactes et approximatives utilisées dans les problèmes d'optimisation Job Shop (Tangour 2007) :

Méthodes exactes	Branch & Bound
	Programmation linéaire
	Programmation dynamique
Heuristiques	LRF (Last Release Time First)
	SPT (Shortest Processing Time)
	LPT (Longest Processing Time)
	MOPNR (Most OperatioNs Remaining or Most Task Remaining)
	MWKR (Most WorK Remaining ou LRT Most Longest Remaining Processing Time)
	LWKR (Least WorK Remaining ou SRT Most Shortest Remaining Processing Time)
	RONDOM
Métaheuristiques	Algorithme Génétique
	Recherche Locale
	Recherche Tabou
	Recuit Simulé
Autres méthodes	Satisfaction des contraintes
	Réseaux des neurones
	Systèmes experts

Table 0.1 Méthodes de résolution de problèmes d'optimisation (Tangour 2007)

3.1 Les méthodes exactes

Parmi ces méthodes, nous citons :

- la méthode de Séparation et d'Évaluation Progressive ou « branch & bound »,
- la programmation linéaire,
- la programmation dynamique.

En général, les méthodes exactes sont utiles dans les cas où le temps de calcul nécessaire pour atteindre la solution optimale n'est pas excessif, ce qui correspond à des problèmes de petite taille.

3.1.1 La méthode branch & bound

Cette méthode consiste à énumérer toutes les solutions d'un problème combinatoire d'une manière intelligente, en éliminant les solutions partielles qui ne mènent pas à la solution que l'on cherche. Une fonction de bornage est utilisée en se basant sur une borne minimale. Cette

fonction sert à réduire l'espace de recherche en écartant toutes les solutions qui sont en dessous de la borne inférieure afin de maintenir que les solutions potentielles.

En effet, la qualité de cette méthode est étroitement liée à la fonction du bornage (sa capacité d'exclure les solutions partielles le plus tôt possible).

Généralement, le déroulement de cette méthode est présenté par une arborescence, dont les racines sont l'ensemble des solutions. Des procédures de bornes inférieures et supérieures sont appliquées à la racine. Si deux bornes sont égales, alors une solution optimale est trouvée. Sinon, l'ensemble des solutions est divisé en deux ou plusieurs sous-problèmes. En appliquant récursivement cette méthode, une arborescence sera générée. Si une solution optimale est trouvée pour un sous-problème (une branche), elle est réalisable, mais pas nécessairement optimale pour le problème de départ.

Une solution réalisable peut être utilisée pour éliminer toute sa descendance : si la borne inférieure d'un nœud dépasse la valeur d'une solution déjà connue, alors nous peuvons dire que la solution optimale globale ne peut pas être incluse dans le sous-ensemble de solutions représentées par ce nœud. La recherche continue jusqu'à ce que tous les nœuds soient explorés, ou éliminés.

3.1.2 La programmation linéaire

Cette méthode s'applique sur les problèmes d'optimisation où la fonction objectif et les contraintes sont toutes linéaires.

Généralement, la programmation linéaire consiste à déterminer *n* variables, $x1, x2, .., xn$ afin de maximiser la fonction objectif linéaire :
$L(x1, \dots, xn) \sum_{i=1}^{n} cixi$ Sous différentes contraintes, comme par exemple les *m* inégalité.

3.1.3 La programmation dynamique

Dans cette technique, toute solution optimale s'appuie elle-même sur des sous-problèmes résolus localement de façon optimale. Concrètement, cela signifie que l'on va pouvoir déduire la solution optimale d'un problème en combinant des solutions optimales d'une série de sous problèmes. Les solutions des problèmes sont étudiées « de bas en haut », c'est-à-dire qu'on calcule les solutions des sous-problèmes les plus petits pour ensuite déduire petit à petit les solutions de l'ensemble.

Par exemple pour optimiser la production de 30 puits à budget donné, on optimise la gestion de 2 puits pour tout budget inférieur ou égal, puis on considère l'ensemble comme un puits unique et on ajoute les puits suivants un par un.

3.2 Les méthodes approximatives

Ces méthodes sont considérées pour les problèmes d'ordonnancement dans lesquels nous ne trouvons pas de solution optimale en un temps raisonnable (Liouane 1998).

Donc, il est intéressant de trouver des méthodes qui fournissent une solution de bonne qualité dans un laps de temps raisonnable. Parmi ces méthodes, les heuristiques.

3.2.1 Les heuristiques

Elles sont basées sur des méthodes empiriques, elles utilisent des règles simples pour optimiser un ou plusieurs critères. Le principe général de cette catégorie de méthodes est d'intégrer des stratégies de décisions pour construire une solution proche de l'optimal tout en essayant d'avoir un temps de calcul raisonnable.

Exemples d'heuristiques :

- **LRF** (Last Release Time First) : Cet algorithme choisit parmi les opérations exécutables celle dont le temps écoulé depuis son apparition est le plus grand. Si aucune tâche n'est disponible, alors un temps libre est généré (Schwiegelshohn 2012).
- **SPT** (Shortest Processing Time) : La prochaine opération ordonnancée est celle dont la durée opératoire est inférieure à celles des autres opérations.
- **LPT** (Longest Processing Time) : La prochaine opération ordonnancée est celle dont la durée opératoire est supérieure à celles des autres opérations.
- **MOPNR** (Most OperatioNs Remaining or Most Task Remaining):
  La prochaine opération ordonnancée est la première opération non encore ordonnancée du Job contenant le plus grand nombre d'opérations non encore achevées.
- **MWKR** (Most WorK Remaining ou LRT Most Longest Remaining Processing Time): La prochaine opération ordonnancée est la première opération non encore ordonnancée du Job dont la somme des durées d'opérations non encore ordonnancées est la plus grande.
- **LWKR** (Least WorK Remaining ou SRT Most Shortest Remaining Processing Time): La prochaine opération ordonnancée est la première opération non encore ordonnancée du Job dont la somme des durées d'opérations non encore ordonnancées est la plus courte.
- **RANDOM** : La prochaine opération ordonnancée est choisie aléatoirement parmi les opérations non encore ordonnancées.

Exemple : La solution S est préparée par les heuristiques suivantes :
S = [RANDOM, LRF, LWKR, RANDOM, LPT, SPT, MOPNR].

3.2.2 Les Méta-heuristiques

Nous avons vu précédemment que les heuristiques se caractérisent par leur simplicité, ce qui les rend rapides en termes de temps d'exécution, mais une méthode heuristique trop simplifiée peut conduire à fournir une mauvaise décision.

L'usage de plusieurs heuristiques afin de créer une méthode plus complexe (Méta - heuristique) permet de réduire considérablement l'espace de recherche. En général, les méta-

heuristiques fournissent une solution quasi-optimale afin de garantir la performance (offrir une solution qui s'approche de l'optimum, tout en réduisant au maximum les coûts). Les Méta-heuristiques ne peuvent pas être classées dans la catégorie de l'intelligence artificielle, puisqu'elles sont guidées par le hasard.

3.2.2.1 Les méthodes par voisinage

Les méthodes par voisinage se basent sur les trois étapes : (F. Carlier 1997) :

1. construire une solution initiale x,
2. déterminer parmi les solutions voisines y de x celles de f(y) minimale,
3. si f(y) < f(x) alors remplacer x par y et aller à la deuxième étape, sinon, retenir x comme meilleure solution.

Ces méthodes ont plusieurs avantages par rapport aux méthodes exactes:

- elles fournissent de bons résultats, tout en réduisant le coût,
- la possibilité de contrôler le temps de calcul (Aydin, et al. 1999) : la qualité de la solution trouvée peut être améliorée progressivement au cours du temps et l'utilisateur a la possibilité d'arrêter l'exécution au moment qu'il aura choisi.

Cette catégorie regroupe plusieurs méthodes, telles que : le Recuit simulé, la Recherche Tabou, qui sont les plus populaires.

3.2.2.2 La Recherche Tabou

Cette technique a été inventée par Glover, elle représente une variante du paradigme de la recherche locale. La composante principale de la recherche Tabou est la fonction de voisinage qui détermine l'efficacité de la recherche. Son principe consiste à améliorer à chaque étape la valeur de la fonction objectif. Une mémoire qui garde les informations sur les solutions déjà visitées est utilisée. Le contenu de cette mémoire représente la liste Tabou qui sert à empêcher l'accès aux dernières solutions visitées. Lorsqu'un optimum local est atteint, il y'a interdiction de revenir sur le même chemin (Fred 1989).

Voici un pseudo code de la Recherche Tabou:

Algorithme Recherche Tabou

Début

Trouver une solution initiale S_0, et posez Z= S_0, i = 0, TL = Φ (TL : liste tabou) ;

Mettre i = i + 1;

Déterminer le sous-ensemble de voisinage $N^*(S_i)$ inclus dans $N(S_i)$ tel que :

$\forall S_{i+1} \in N^*(S_i)$: $(S_i, S_{i+1}) \notin$ TL ; ((S_i, S_{i+1}) : le mouvement de S_i à S_{i+1}) ;
Remplacer S_i par S_{i+1} tel que S_{i+1} est la meilleur solution de $N^*(S_i)$;

Mettre à jour TL ;

Si la condition d'arrêt n'est pas vérifiée, alors aller à 3.

Fin

3.2.2.3 Le recuit simulé

Inspiré du recuit physique, ce processus est utilisé en métallurgie pour améliorer la qualité d'un solide et cherche un état d'énergie minimale qui correspond à une structure stable du solide. Ainsi, pour qu'un métal retrouve une structure proche du cristal parfait, nous portons celui-ci à une température élevée, puis nous le laissons refroidir lentement de manière à ce que les atomes aient le temps de s'ordonner régulièrement (Kirkpatrick 1984).

En optimisation (Collette et Siarry 2002) (Hao, Galinier et Habib 1999), le processus du recuit simulé répète une procédure itérative qui cherche des configurations de coût plus faible tout en acceptant de manière contrôlée des configurations qui dégradent la fonction de coût. Ainsi, si une amélioration du critère est constatée, le nouvel état est retenu, sinon une diminution ΔE du critère est calculée et le nouvel état est retenu avec une probabilité $P = e^{-\Delta E}/_{T}$.

Donc, à partir d'une solution initiale S_0, à chaque nouvelle itération, un voisin $S' \in N(s)$ de la configuration courante S est généré de manière aléatoire. Selon le cas, ce voisin sera soit retenu pour remplacer celle-ci, soit rejeté.

- Si ce voisin est de performance supérieure ou égale à celle de la configuration courante ($f(s') \leq f(s)$), il est systématiquement retenu ;
- Dans le cas contraire, s'est accepté avec une probabilité P qui dépend de deux facteurs :

- De l'importance de la dégradation $\Delta f = f(s') - f(s)$; les dégradations les plus faibles sont plus facilement acceptables.
- D'un paramètre de contrôle T (la température), une température élevée correspond à une probabilité plus grande d'accepter des dégradations (Aydin, et al. 1999).

Généralement, la température est contrôlée par une fonction décroissante qui définit un schéma de refroidissement. Il existe trois schémas :

- Réduction par palier : la température est maintenue constante pendant un certain nombre d'itérations, et décroit ainsi par palier.
- Réduction continue : la température est modifiée à chaque itération.
- Réduction non monotone : la température décroit à chaque itération avec des augmentations occasionnelles.

Algorithme recuit simulé

Algorithme

Begin

Déterminer la solution initiale s_0 et la température initiale T_0 ;
Poser $Z = s_0$ et $Ti = T_0$;
Calculer $f(s_0)$;
Répéter pour chaque itération i / $i = 1, \ldots, n$ (ou jusqu'à ce que T soit proche de 0) :

a. Choisir $s' \in N(s_i)$;
b. Calculer $\Delta f = f(s') - f(s_i)$;

c. **If** $\Delta f < 0$ **Then** $Z = s'$; // la nouvelle solution est plus bonne que l'ancienne
d. **Else**
begin
Tirer P de [0,1]
If $P \leq e^{\Delta f/T}$: **Then** $Z = s'$
Else s' est rejetée
End
e. Calculer $T = \alpha T$
Retourner Z
End.

3.2.2.4 Les algorithmes génétiques

Les algorithmes génétiques tentent de reproduire l'évolution naturelle d'individus en respectant la loi de survie énoncée par Darwin. Le principe de base des algorithmes génétiques développés initialement par Holland pour répondre à des besoins spécifiques en biologie, ont été rapidement appliqués pour résoudre et avec succès les problèmes d'optimisation combinatoire en recherche opérationnelle et les problèmes d'apprentissage dans le domaine de l'intelligence artificielle (Goldberg 2008). Parmi les premières applications des algorithmes génétiques aux problèmes combinatoires tels que les problèmes d'ordonnancement: (Lawrence 1985), (Nakano et Yamada 1991), (Yamada et Nakano 1992).

A. Principe

Dans le cadre des problèmes d'optimisation combinatoires, un individu dans une population représente une solution parmi l'ensemble de solutions possibles pour un problème particulier. L'application d'un algorithme génétique à un problème particulier nécessite cinq éléments suivants:

- **Le codage des solutions (individus)**

Associe à chacun des points de l'espace de solution une structure de données. Elle vient généralement après une phase de modélisation mathématique du problème traité. La qualité du codage conditionne le succès de l'algorithme.

- **Une méthode de génération de la population initiale**

La population initiale qui servira de base pour les générations futures doit être non homogène.

- **Une fonction à optimiser**

Cette fonction sert à évaluer chaque individu, elle retourne une valeur réelle nommée : « fitness », qui sera utilisée dans le calcul de probabilité de sélection d'un individu.

- **Les opérateurs génétiques**

La puissance des algorithmes génétique réside dans leurs opérateurs génétiques (croisement et mutation). En effet, ces opérateurs permettent d'établir un balayage partiel de l'espace de recherche, ce qui réduit son temps de réponse.

Le croisement est considéré comme l'opérateur génétique le plus important. Son rôle consiste à recomposer les gènes d'individus de la population afin de balayer l'espace de solutions.

Le deuxième opérateur génétique (Mutation) est moins important par rapport au premier opérateur. Il sert à introduire des petites perturbations au niveau de la population dans le but d'éviter une convergence prématurée de l'algorithme génétique vers un ou plusieurs optimums locaux.

- **Algorithme général**

Les étapes des algorithmes génétiques peuvent varier d'un problème à un autre. Mais, ils se basent sur le même principe. La figure suivante (**Figure 0.4**), illustre le principe de base des algorithmes génétiques (Barnier et Brisset 1999) :

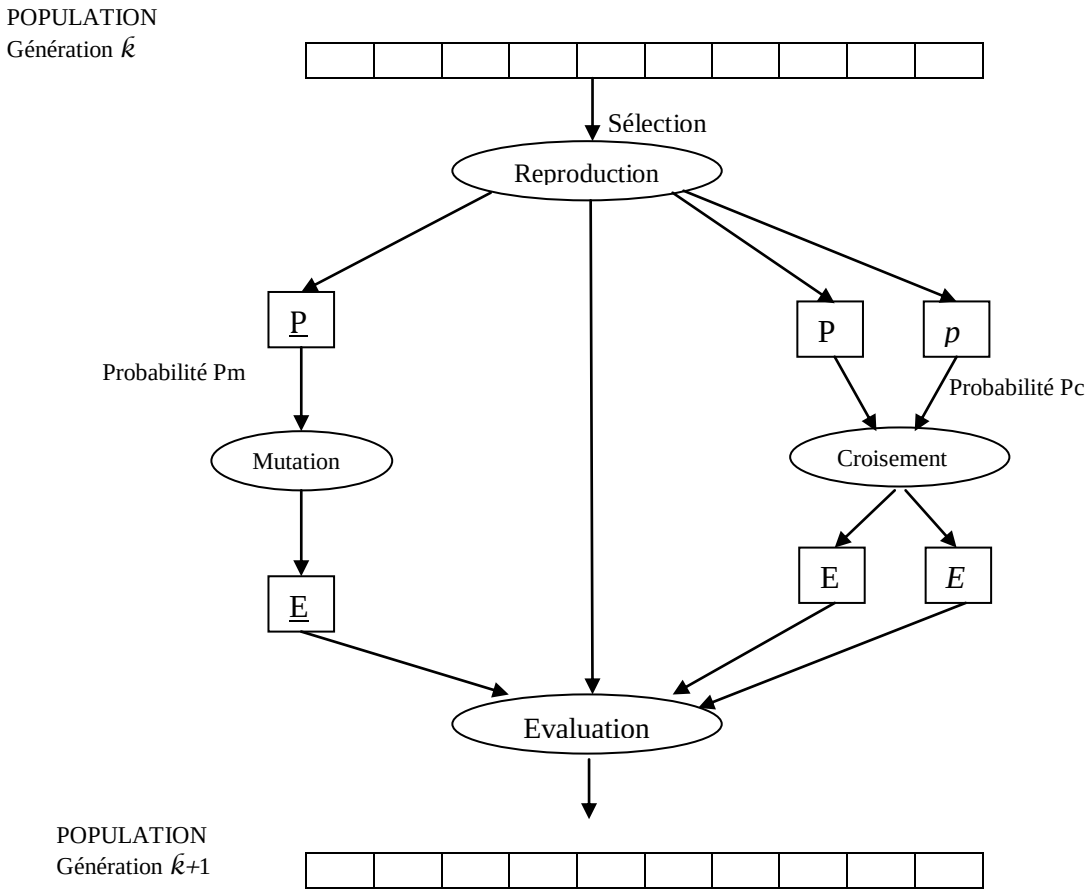


Figure 0.4 Principe général des algorithmes génétiques

L'algorithme suivant résume le principe de base de l'algorithme génétique :

Algorithme génétique

Begin

Génération d'une population initiale de k individus ;

Repeat

1. Evaluation de la fonction objectif f de chaque individu ;
2. Sélection des meilleurs individus ;
3. Croisement entre 2 individus: génération 2 enfants issus de deux parents sélectionnés ;
4. Mutation : transformation aléatoire des gènes de certains individus de la population ;

Until la satisfaction de la condition d'arrêt.

Retourner la meilleure solution.

End.

B. Codage

Les algorithmes génétiques sont appliqués sur une population d'individus, chacun de ces derniers est codé par un *chromosome* ou génotype. Donc, une population est présentée par un ensemble de chromosomes. Le processus de codage d'individus consiste à trouver une structure de données pour les individus. Plusieurs types de codage existent, tels que : le codage binaire, par permutation de valeurs entières et par valeur, etc. (Fontanili 1999).

B.1 Le codage binaire

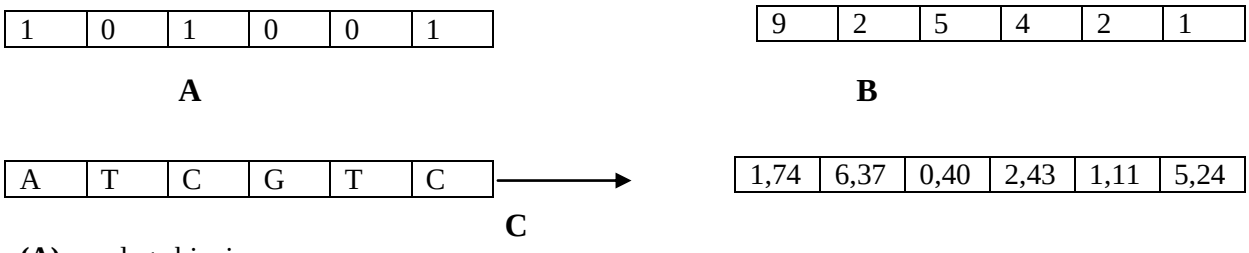
Les premiers algorithmes génétiques ont utilisé le codage binaire. Un chromosome est représenté par une chaine binaire (chaine de bits) précisant l'information nécessaire à la description d'un individu.

B.2 Le codage par valeurs entières

Chaque gène est codé par une valeur entière. Donc, un chromosome est représenté sous forme d'une chaine d'entiers. Ce codage est bien adapté à l'optimisation des problèmes industriels réels (Fontanili 1999).

B.3 Le codage par valeur

Dans ce type de codage (**Figure 0.5**), chaque gène est codé par une valeur qui appartient à un ensemble fini ou infini. Ces valeurs sont liées au problème à résoudre.



(A) : codage binaire,
(B) : codage par permutation de valeurs entières,
(C) : codage par valeur.

Figure 0.5 Exemples de codage par valeurs

C. Les opérateurs génétiques

Nous distinguons trois opérateurs génétiques de reproduction : l'opérateur de sélection, l'opérateur de croisement et celui de mutation.

C.1 Le croisement

L'opérateur de croisement est un opérateur binaire, il consiste à choisir une paire d'individus afin de recombiner les deux chromosomes sélectionnés dans le but de produire deux nouveaux chromosomes enfants portants les meilleures caractéristiques des deux chromosomes parents. Il existe plusieurs stratégies de croisement :

- **Le croisement à un point simple**

Dans ce type de croisement (**Figure 0.6**), nous définissons un point de coupure au niveau des deux chromosomes Parents, ensuite, chaque chromosome enfant est construit par l'une des deux chaines terminales de chacun des deux chromosomes Parents (voir la figure 2.6) :

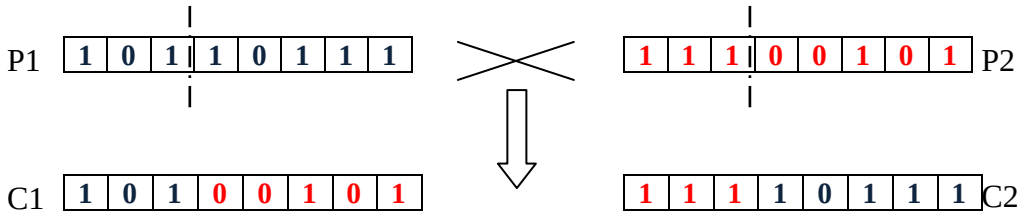


Figure 0.6 Exemple d'un croisement à point simple

- **Le croisement Multipoints**

Le croisement est dit multipoints, dans le cas où le découpage du chromosome peut se faire non seulement en 2 sous-chaînes, mais en 3 ou 4 sous-chaînes, etc. Donc, ce croisement permet d'échanger plusieurs segments entre deux chromosomes.

Le croisement bipoints (**Figure 0.7**) est un cas particulier du croisement multipoints dont deux points de croisement sont choisis afin d'échanger les deux segments des deux parents déterminés par ces deux points

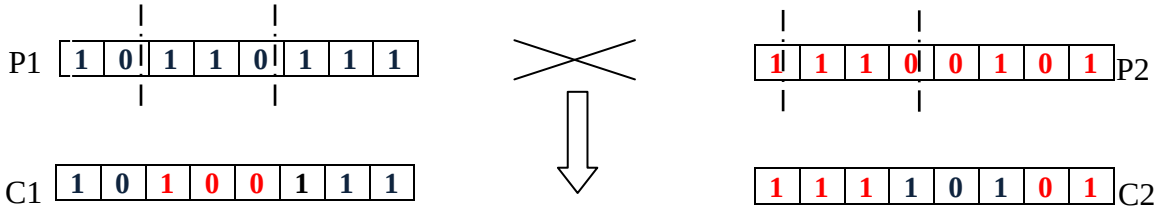


Figure 0.7 Exemple d'un croisement bipoints

- **Le croisement uniforme**

Nous trouvons plusieurs approches liées à ce type de croisement (**Figure 0.8**), parmi ces approches les plus connues celles qui utilisent un vecteur masque. Le masque est un vecteur de la même taille que celle du chromosome, il contient une suite de 0 et de 1 déterminée aléatoirement. Il sert à déterminer de quel parent seront repris les gènes.

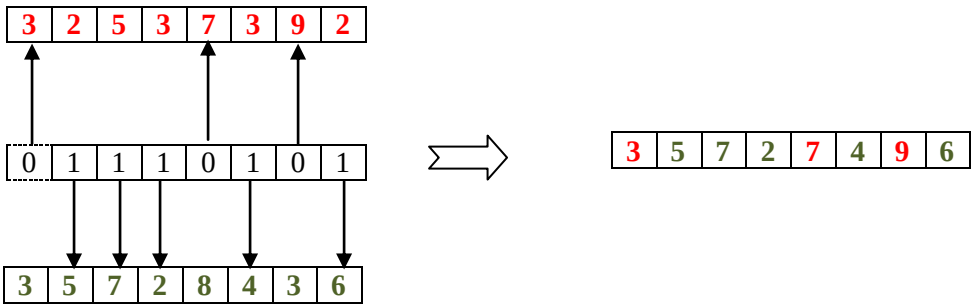


Figure 0.8 Exemple de croisement uniforme

Le croisement uniforme (**Figure 0.8**) peut être utilisé dans le cas où les gènes sont soumis à certaines restrictions du codage, donc ces restrictions sont imposées aussi aux chromosomes enfants.

- **Le croisement multi-parental**

Consiste à combiner les gènes de plus de deux parents en utilisant par exemple la méthode du simplexe pour orienter la recombinaison (Fontanili 1999).

C.2 La Mutation

L'opérateur de mutation, assure le brassage et la recombinaison des gènes parentaux, permettant de diversifier la future population afin d'éviter une convergence rapide vers une solution optimale localement.

Étant aléatoire, cet opérateur agit selon une probabilité *Pc* fixée par l'utilisateur en fonction du problème à optimiser (Mesghouni 1999).

Il existe plusieurs stratégies de mutation:

- **La mutation uni-point**

Ce type de mutation se fait par altération d'une seule valeur sur le chromosome.

- **La mutation bipoints et multipoints**

Cette mutation se fait par altération de plusieurs valeurs sur le chromosome.

- **La mutation par valeurs**

La mutation par valeur se fait par transformation d'une valeur donnée en une autre valeur déterminée, sur tous les gènes du chromosome.

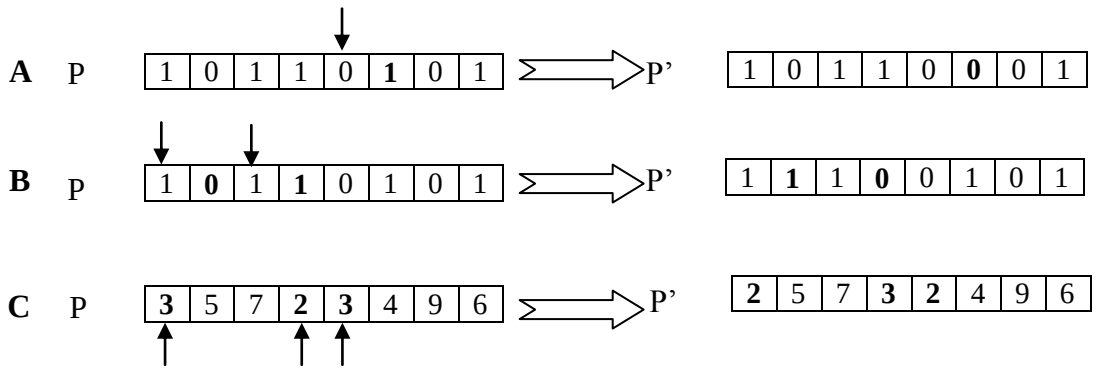


Figure 0.9 Exemple d'opérateurs de mutation.

C.3 La sélection

L'opérateur de sélection a pour objectif de choisir les individus qui vont pouvoir survivre et/ou se reproduire pour transmettre leurs caractéristiques à la génération suivante. La sélection se base sur le principe de conservation des individus les mieux adaptés et d'élimination des moins adaptés (Hao, Galinier et Habib 1999). Il existe plusieurs stratégies de sélection, parmi ces stratégies les plus importantes, nous citons (Haupt et Haupt 2004):

- **La sélection par la roue de fortune (roulette Wheel)**

C'est une roue décomposée en plusieurs secteurs (**Figure 0.10**), où chacun correspond à un chromosome de population. La superficie de chaque secteur est relative à la valeur de la fonction objectif. Donc, plus la force d'adaptation de l'individu est importante, plus la superficie de son secteur est importante.

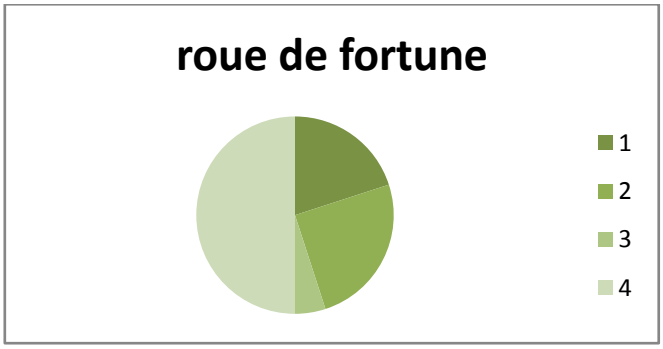


Figure 0.10 Sélection par la « roue de fortune »

Donc, la probabilité avec laquelle chaque individu sera sélectionné est:

$$P(\text{i}) = f(d(ci))/\sum_{i=1}^{N} f(d(ci)) \qquad \text{Equ. 0-1}$$

Où, $f(d(ci))$ est la fitness du chromosome i.

- **La sélection par rang (Ranking)**

Dans cette stratégie de sélection, les individus d'une population sont triés de 1 à N en fonction de leur fitness. Ainsi, le plus mauvais chromosome aura le rang 1, le meilleur chromosome aura le rang N. Nous associons à chaque chromosome i une probabilité Pi d'être sélectionné, la sélection sera proportionnelle à leurs rang dans la liste de la population :

$$Pi = Ri/\sum_{i=1}^{N} Ri \qquad \text{Equ. 0-2}$$

L'exemple suivant illustre la méthode de sélection par rang (**Table 0.2**) :

Chromosomes	**1**	**2**	**3**	**4**	**5**	**6**	**Totale**
Probabilités de fitness	87 %	7 %	5 %	2 %	1 %	94 %	100 %
Rangs	5	4	3	2	1	6	21
Probabilités de rang	24 %	19 %	15 %	5 %	9 %	29 %	100 %

Table 0.2 Exemple de sélection par rang

Ensuite, cette probabilité sera comparée avec un nombre aléatoire afin de prendre une décision. Le principe de cette technique est illustré dans l'algorithme suivant :

Algorithme de sélection par rang

Classer les individus de la population courante selon leurs valeurs de fitness (ordre croissant),
For i = 1 To *N* **Do**
  Calculer la probabilité de sélection *Pi*
  Générer un nombre x appartenant à l'intervalle [0, 1]
  If x < *Pi* **Then**
    Sélectionner l'individu *i*
  EndIf
EndFor

- **La sélection steady-state**

Le principe de cette stratégie de sélection consiste à suivre les étapes suivantes au niveau de chaque génération :

- Sélectionner quelques chromosomes parmi ceux qui ont le meilleur coût, afin de créer des chromosomes fils (croisement et mutation) ;
- Remplacer les chromosomes les plus mauvais par les nouveaux chromosomes ;
- Les chromosomes restants surviennent à la nouvelle génération.

Donc, le principe de cette sélection consiste à faire reproduire les meilleurs, et exclure les mauvais, en ne donnant aucune chance aux mauvais de survivre.

- **La sélection par tournoi**

Cette stratégie consiste à tirer des paires d'individus d'une population de taille *N*. Au niveau de chaque paire, un individu sera sélectionné selon une probabilité dite de *victoire* de plus fort. Cette probabilité représente la chance pour que l'individu soit sélectionné.

Il existe d'autres variétés de sélection par tournoi, telles que : le tournoi à trois, où la sélection se fait par trois individus dont un seul sortira vainqueur avec une certaine probabilité de victoire.

- **L'élitisme**

Il se peut après l'application des opérateurs génétiques de croisement et de mutation qu'une partie des meilleurs chromosomes sera perdue. La stratégie d'élitisme est utilisée afin d'éviter ce problème. Son principe consiste à copier un ou plusieurs des meilleurs chromosomes dans la nouvelle génération, ensuite, nous utilisons l'algorithme de reproduction usuel pour générer le reste de la population.

D. Paramètres de dimensionnement

Il existe plusieurs paramètres prédéfinis qui jouent un rôle crucial dans la performance de chaque algorithme génétique. Parmi ces paramètres de base (nous pouvons trouver d'autres paramètres qui dépendent du problème traité):

- La taille de la population *N* : plus la valeur de *N* est grande (la taille), plus le temps de recherche est important. Par contre, si la taille est trop petite, l'algorithme risque de converger trop rapidement vers une mauvaise solution.
- La probabilité de croisement *Pc* : la valeur de cette probabilité reflète l'intensité des changements appliqués sur la population. Généralement, la probabilité de croisement est comprise entre 0,5 et 0,9.
- La probabilité de mutation *Pm* : un taux élevé de *Pm* risque de converger vers une solution sous-optimale et à la perte de la population originale, c'est pour cette raison que la probabilité de mutation est généralement faible.
- Le nombre de générations peut être défini comme une condition d'arrêt.

4. L'OPTIMISATION MULTI-OBJECTIF ET LES ALGORITHMES GENETIQUES

Ces algorithmes sont très adaptés pour la résolution des problèmes d'optimisation multi-objectif.

4.1. La méthode M.O.G .A (Multiple Objective Genetic Algorithm)

MOGA est une extension de l'algorithme génétique classique (Fonseca et Fleming 1995). La différence principale entre GA et MOGA réside dans l'assignement d'une valeur de fitness à chaque individu. Le reste de l'algorithme est similaire à l'algorithme génétique classique.

Le rang d'une solution i est donné par :

$$r_i = 1 + P_i \qquad \text{Equ. 0-3}$$

P_i est le nombre de solutions qui dominent la solution i.
Pour calculer la fitness 'efficacité' de la solution i, nous suivons les étapes suivantes :
1. classer les solutions selon leur rang,
2. affecter une efficacité à chaque solution en interpellant à partir du meilleur rang au plus mauvais. Le principe de MOGA est décrit de la manière suivante :

Algorithme M.O.G.A

Initialisation de la population
Evaluation des fonctions objectif
Affectation d'un rang selon la dominance
Affectation d'une efficacité à partir du rang
For i = 1 **to** Max_ génération
  Sélection aléatoire proportionnelle à l'efficacité
  Croisement
  Mutation
  Evaluation des fonctions objectif
  Affectation d'un rang selon la dominance
  Affectation d'une efficacité à partir du rang
End$_{\text{For}}$

L'inconvénient de cette méthode réside dans le manque de diversité dans la représentation des solutions.

4.2. La méthode NSGA (Non Dominated Sorting Genetic Algorithm)

Cette méthode s'articule sur le même principe de la méthode MOGA. En revanche, l'efficacité des individus est calculée autrement.

Chaque solution est testée pour sa domination dans la population. Le rang d'une solution *i* à la génération t est donné par :

$$r_i = 1 + P_i \qquad \text{Equ. 0-4}$$

tel que P_i^t est le nombre d'individus qui dominent la solution *i*.

Donc, les solutions non dominées sont assignées au rang 1. Apres avoir assigné un rang à chaque individu, une fitness est assignée à chaque solution en se basant sur son rang. Les rangs sont triés selon un ordre croissant ensuite, une fonction est utilisée pour assigner la fitness N pour le meilleur rang et 1 pour le mauvais.

Afin de maintenir la diversité parmi les solutions non dominées, une technique nommée *niching* est utilisée en calculant *nich count* pour chaque solution (Eq. 2.5).

$$nc_i = \sum_{j=1}^{\mu(r_i)} Sh(d_{ij}) \qquad \text{Equ. 0-5}$$

La fonction *Shared function* $Sh(d_{ij})$ est calculée en utilisant la valeur de la fonction objective comme une distance métrique :

$$Sh(d_{ij}) = \begin{cases} 1 - \left(\frac{d_{ij}}{\sigma_{share}}\right)^{\alpha} & if\ d_{ij} \leq \sigma_{share} \\ 0 & else \end{cases} \qquad \text{Equ. 0-6}$$

σ_{share} est le paramètre de partage qui veut dire la distance maximale entre deux solutions avant qu'elles soient considérées d'être dans la même niche.
α est un facteur d'échelle inferieur ou égale à 1.
d_{ij} est une distance normalisée entre deux solutions i et j dans un rang, elle est calculée en utilisant :

$$d_{ij} = \left[\sum_{k=1}^{M} \left(\frac{{f_k}^i - {f_k}^j}{{f_k}^{max} - {f_k}^{min}}\right)^2\right]^{\frac{1}{2}} \qquad \text{Equ. 0-7}$$

${f_k}^{max}$ et ${f_k}^{min}$ sont la valeur maximale et minimale de la fonction objective *k*.

La valeur de la fitness partagée '*The shared fitness'* est calculée par la division de la valeur de fitness assignée à une solution par le compteur de niche '*the niche count*'. La valeur de la fitness partagée '*efficacité'* d'une solution résidant dans une région moins bondée a une meilleure fitness partagée qui est calculée par la formule suivante :

$$F'_j = \frac{F_j}{nc_j} \qquad \text{Equ. 0-8}$$

La formule 2.8 permet de réduire la valeur de fitness de chaque solution.

Cette procédure est répétée jusqu'à ce que tous les rangs soient traités. Apres, les opérateurs sélection, croisement et mutation sont appliqués afin de créer une nouvelle population.

4.3. La méthode SPEA (Strenght Pareto Evolutionary Algorithm)

Zitzler et Thiele (Zitzler et Thiele 2000) ont implémenté la méthode SPEA qui est considérée comme une illustration d'un algorithme évolutionniste élitiste. SPEA préserve un archive externe contenant les meilleurs fronts de compromis trouvés durant la recherche.

Algorithme SPEA

Initialiser la population P_0 et créer l'archive externe vide *P'*

Mise à jour de *P'* à partir des individus non dominés de P_0

Tant que critère d'arrêt non vérifié **Faire**

Calculer la valeur d'adaptation pour les individus de P + *P'*

Sélection à partir de P_t + *P'* en fonction de la valeur d'adaptation

Croisement

Mutation

Mise à jour de *P'* à partir des individus non dominés de P.

Fin$_{\text{Tant que}}$

Une version améliorée de SPEA nommée SPEA-2 a était proposée par (Zitzler, Laumanns et Thiele 2001).

4.4. La méthode PAES (The Pareto Archived Evolution Strategy)

Cette méthode était crée par (Knowles et Corne 1999). Elle se base sur une stratégie evolutioniste simple de la recherche locale. Un archive externe des solutions non dominées est maintenu afin de garantir la diversité des solutions trouvées.

5. CLASSIFICATION DES APPROCHES DE RESOLUTION DES PROBLEMES JOB SHOP FLEXIBLE

L'avantage des ateliers Job-Shop flexibles réside dans leur capacité de reconfiguration afin de traiter une grande variété d'opérations. En revanche, cet avantage peut disparaitre. C'est-à-dire la considération de flexibilité dans l'ordonnancement du Job-Shop classique va gravement augmenter la complexité du problème, qui est difficile à résoudre.

Donc, le problème d'ordonnancement des ateliers Job Shop flexibles peut être décortiqué en deux sous problèmes : *assignement* des machines aux opérations et séquencement des opérations au niveau de chaque machine.

Les problèmes d'ordonnancement des ateliers de type Job Shop Flexibles peuvent être traités en utilisant soit, une approche intégrée, soit une approche hiérarchique.

1. *Intégrée* :
   Dans cette approche, les deux sous problème sont traités intégralement, c'est-à-dire sans diviser le problème initial. Enormément de travaux sont réalisés pour la résolution intégrée des problèmes d'ordonnancement (Jones, Rabelo et Sharawi 1998).
2. *Hiérarchique* :
   Contrairement à l'approche intégrée, l'approche hiérarchique traite le problème en deux phases en divisant le problème initial en deux sous problèmes plus simples, dans le but de réduire la complexité du problème.

La plupart des méthodes récentes pour la résolution des problèmes d'ordonnancement JSF sont des approches méta-heuristiques qui permettent de trouver une solution de bonne qualité dans un laps de temps raisonnable, parmi ces travaux :

L'une des premières approches hiérarchiques pour JSF a été proposée par Brandimarte (Brandimarte 1993), basée sur la recherche Tabou. La coordination entre les deux phases a été prise en considération en utilisant un flux d'information bidirectionnelle.
Deux années plus tard, Pauli (Paulli 1995) a proposé deux phases pour ordonnancer un système de production manufacturier, dans lequel le nombre de jobs qui peuvent être exécutés simultanément est limité. Dans la première phase, les machines sont assignées aux opérations par les règles de priorités

Récemment, plusieurs méthodes basées sur les approches métaheuristiques ont été proposées pour JSF (Mastrolilli et Gambardella 2000), (Kacem, Hammadi et Borne 2002), (Zribi, et al. 2007), (Zribi, El-Kamel et Borne 2008), (Raich, Nikumbh et Patil 2012), (Sridhar, Victor Raj et Chandra sekar 2013), (Ziaee 2014), (Yiyong, Wei et Shigeru 2017).

La résolution des problèmes d'ordonnancement des ateliers Job Shop totalement flexibles avec le temps de transport reste limitée. En plus, Ces études ont été appliquées sur des benchmarks théoriques, qui ignorent des contraintes réelles, telles que le temps de transport.

Dans le chapitre suivant, nous présentons une méthode d'ordonnancement du Job shop flexible avec le temps de transport. L'optimisation est mono objectif (durée de production). Par contre, dans le chapitre 4, une optimisation multi-objectif est proposée avec une prise en considération de la consommation de l'énergie.

6. CONCLUSION

Dans ce chapitre, nous avons vu les concepts liés aux problèmes d'optimisation combinatoires ainsi que ses différentes méthodes de résolution. Ces méthodes sont regroupées en deux classes principales : les méthodes exactes et les méthodes approchées.

Les méthodes exactes sont utiles dans le cas où le temps de calcul nécessaire pour atteindre la solution optimale n'est pas excessif, c'est généralement le cas des problèmes de petite taille. En revanche, pour les problèmes de grande taille, les méthodes approchées offrant des solutions avec un temps de réponse acceptable deviennent plus utiles.

Pour notre problème Job Shop Flexible, nous avons choisi les algorithmes génétiques qui font parti des méthodes approximatives pour l'ordonnancement des opérations, en prenant en considération le temps de transport entre les machines. Ces algorithmes sont bien adaptés aux problèmes combinatoires. La recherche tabou est intégrée au niveau de l'algorithme génétique pour améliorer les performances.

CHAPITRE 3 : LE PROBLEME JOB SHOP FLEXIBLE AVEC TRANSPORT

1. INTRODUCTION

Dans ce chapitre, nous nous intéressons au problème job shop flexible avec transport. Nous proposons un algorithme génétique hybride (HGA) pour résoudre le problème considéré. Une Recherche Tabou améliorée est proposée basée sur une nouvelle fonction de voisinage, afin d'améliorer les performances de l'algorithme génétique.

Les solutions sont évaluées sur la base de la durée totale d'achèvement de tous les jobs (makespan), en prenant en considération le temps de transport entre les machines.

Ce chapitre est organisé de la manière suivante : un état de l'art est présenté sur les travaux précédents qui ont porté sur l'ordonnancement des ateliers de type job shop. Dans la deuxième section, nous donnons une formalisation mathématique du problème de l'ordonnancement job shop flexible avec transport. Afin de simplifier notre approche, une illustration du modèle d'optimisation est donnée, tout en expliquant comment la recherche tabou proposée a été intégrée dans l'algorithme génétique et la contribution de cette hybridation. Dans les deux sections qui suivent, nous détaillons la recherche Tabou et l'algorithme génétique utilisés pour l'optimisation. Nous clôturons ce chapitre par une présentation de la cellule de production utilisée pour tester notre approche et les résultats obtenus.

2. ETAT DE L'ART

Les ateliers Job shop et Flow shop ont fait l'objet de plusieurs études de modélisation pour la résolution des problèmes d'ordonnancement avec temps de transport. King et al (King, Hodgson et Chafee 1993) développent un algorithme Branch and Bound pour la résolution du Flow shop avec un seul transporteur. La première modélisation du job shop avec transport a été introduite par Knust (Knust 1999). Ensuite, ce problème a été étudié avec un seul robot, par (Hurink et Knust 2002). Une modélisation sous forme de graphe disjonctif a été proposée par Hurink et al (Hurink et Knust 2005). Ce graphe est constitué de deux types de nœuds : des nœuds reflétant des opérations effectuées sur machines et des nœuds représentant des opérations de transport.

Strusevich (Strusevich 1999) a introduit un décalage connu et déterministe, entre la fin d'une opération et le début de l'opération suivante dans le même job. Ce décalage entre deux opérations consécutives est nommé temps de transport. Récemment, L'étude du Flow shop avec transport a fait l'objet d'une attention particulière (Larabi 2010) dont nous trouvons un état de l'art proposé par Crama et al (Crama, et al. 2000). Ces études considèrent plus particulièrement les problèmes d'ordonnancement avec unique transporteur avec la prise en compte du transport à charge et à vide. D'autres auteurs (Nowicki 1999) (Brucker, Heitmann et Brucker 2003) ont proposé un graph disjonctif avec une capacité du stock limitée, pour résoudre le problème d'ordonnancement du Flow shop avec transport. Pour le job shop sous la contrainte de capacité de stockage, Caumond et al (Caumond, et al. s.d.) ont proposé un

modèle linéaire mixte pour 5 jobs et 111 opérations. Plusieurs auteurs ont étudié le Job shop avec transport et plusieurs robots (Lacomme, Larabi et Tchernev 2007), (Lacomme et Larabi 2008), (Lacomme, Larabi et Tchernev 2013).

Burgy et al (Bürgy et Gröflin 2016) ont proposé une formulation pour le problème du Job shop avec un système de transport nommé rail-bound. Ces auteurs ont développé une recherche locale pour résoudre le problème. Zeng et al (Zeng, Tang et Yanb 2014) ont étudié le même problème mais avec un nombre limité de transporteurs. Ces auteurs ont proposé une heuristique à deux niveaux.

Un algorithme nommé iterative greedy insertion technique a été propose par Bekar et al pour le job shop flexible avec transport (Bekkar, et al. 2016). Ces auteurs ont travaillé sur AIP-PREMICA (Trentesaux, et al. 2013), reflétant une cellule réelle du job shop flexible. Cette technique est très populaire pour résoudre les problèmes d'optimisation combinatoires (Talb 2009). Elle fonctionne d'une manière récursive et destructive. Chaque itération écrase sa précédente, jusqu'à ce que l'optimum local soit trouvé.

Généralement, le problème d'ordonnancement du Job shop sous des contraintes de transport repose sur un ensemble de robots identiques, transportant les jobs entre machines. En revanche, dans le job shop flexible, le transfère peut demander plus de distance (Afsar, et al. 2016). Ainsi, il est plus probable que le transfère est effectué par des véhicule ayant une plus grande capacité.

Nous proposons un algorithme génétique hybride, basé sur la recherche tabou pour la cellule de montage des pièces AIP-PREMICA (Trentesaux, et al. 2013), qui reflète une cellule réelle du problème job shop flexible avec transport. Une fonction de voisinage est proposée pour la recherche tabou.

Dans la section suivante, nous présentons une formalisation mathématique du problème job shop flexible avec transport.

3. FORMULATION DU PROBLEME

Le problème consiste à organiser l'exécution de N Jobs sur M machines. Chaque Job J_j est composé d'un nombre d'opérations ordonnées a_j (contrainte de précédence). Chaque opération doit être exécutée sur une machine. L'exécution de l'opération $O_{j,i,k}$ (la $i^{ème}$ opération du Job J_j exécutée sur la machine M_k) rend cette machine indisponible pour les autres opérations pendant une durée $P_{j,i,k}$ (contrainte de ressource). Donc, une table P_{JSF} des durées de traitement peut être associée à chaque PJSF (problème Job-shop flexible).

En plus des contraintes classiques, nous prenons en considération le temps de transport entre les machines. Afin d'achever un job J_j, ce dernier est transporté à travers un convoyeur automatiques d'une machine M_{m1} à une autre machine M_{m2} dans un temps $T_{j,\ m1-\ m2}$. Notre objectif est de minimiser le temps de transport total de tous les jobs.

Un exemple du Job Shop Flexible est présenté dans la Table 3.1 :

		M_1	M_2	M_3	M_4
J_1	$O_{1,1}$	4	6	8	1
	$O_{1,2}$	7	8	11	6
	$O_{1,3}$	5	1	2	1
J_2	$O_{2,1}$	8	3	6	4
	$O_{2,2}$	6	5	9	3
J_3	$O_{3,1}$	7	9	5	6
	$O_{3,2}$	4	4	2	8
	$O_{3,3}$	3	4	2	3

Table 0.1 Exemple de PJSF

Le problème JSF présente deux difficultés : la première est d'assigner une machine M_K (sélectionnée parmi les M machines) à chaque opération $O_{j,i.}$ La seconde consiste à calculer la date de début $t_{j,i}$ et la date de fin $tf_{j,i}$ de chaque opération $O_{j,i}$. L'objectif considéré est de minimiser la durée totale d'achèvement de tous les jobs (makespan):

Le *makespan (cmax)*:

$$Cmax = \max_{1 \leq j \leq N} tf a_j$$

$$t_{j,i,k} = max\left(tf_{j,i-1,k'} + Tr_{kk'} \ , \ tf^{-1}{}_{j,i}\right) \qquad \text{Equ. 0-1}$$

Nous notons :

- O_{nj} : le nombre total des opérations du job j.
- $P(O_{j,i,k})$: Le temps d'exécution de l'opération $O_{j,i}$ par la machine M_k.
- $t\ O_{j,i,k}$ et $tf\ O_{j,i,k}$: la date de début et la date de fin de l'opération $O_{j,i}$ exécutée par M_k.
- $O^{-1}{}_{j,i,k}$ et $O^{+1}{}_{j,i,k}$: les deux opérations qui précèdent et succèdent $O_{j,i}$ dans la même machine M_k.
- $O_{j,i-1,k}$ et $O_{j,i+1,k}$: les deux opérations qui précédent et succèdent $O_{j,i}$ dans la même gamme opératoire du Job j.
- $OC_{j,i,k}$ est une opération critique, si la durée d'achèvement du job de cette opération égal à max $tf_j = C_{max}$.
- $Tr_{kk'}$ le temps de transport de la machine k vers la machine k'.

Le Job Shop Flexible ne prend pas en compte le temps de transport entre les machines. Les jobs ne sont pas considérés naissant sur les machines. Donc, en plus des contraintes de

précédence et ressource (machines), s'ajoute une contrainte de ressource de transport, souvent des robots transporteurs connus sous l'appellation AGV (Automated Guided Vehicule).

Dans la section suivante, nous présentons notre approche d'optimisation pour le job shop flexible avec transport.

4. L'APPROCHE D'OPTIMISATION PROPOSEE

Dans cette partie, nous répondrons à trois questions : 1. Pourquoi nous avons combiné l'algorithme génétique avec la recherche tabou ? 2. Comment cette combinaison est faite ? et quelle est la contribution proposée ?

La force des algorithmes génétiques réside dans leur capacité d'explorer intelligemment l'espace de solutions. En revanche, une solution optimale pourrait être cachée dernière une solution (une solution voisine). Dans cette situation, l'algorithme génétique devient insuffisant pour atteindre ces solutions voisines qui pourraient être meilleures que la solution précédente ou même la meilleure solution. Pour faire face à ce problème, la recherche Tabou est parfaitement adaptée pour explorer ces solutions voisines.

Après avoir appliqué les deux opérateurs génétiques (croisement et mutation) **Figure 0.1**, une partie de la population serait sélectionnée pour appliquer la recherche tabou. Cette dernière renforce la qualité des solutions et contribue à sa diversification. Ensuite, les meilleures solutions seront sélectionnées pour la prochaine itération.

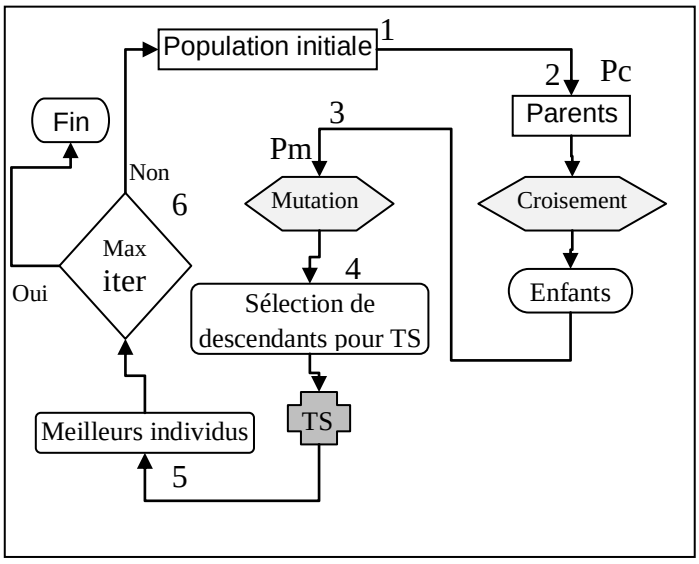


Figure 0.1 Intégration de la recherche tabou (TS) dans l'algorithme génétique (AG)

Selon la **Figure 0.1**, l'approche d'optimisation proposée pourrait être décomposée en plusieurs étapes :

1. Génération de la population initiale

2. Sélection des parents et application du croisement
3. Sélection des chromosomes et application de mutation
4. Sélection des descendants pour la recherche Tabou (TS)
5. Sélection des chromosomes survivants pour la génération suivante
6. Aller à l'étape 2 si le nombre d'itération maximum n'est pas atteint.

Les parents sont sélectionnés en prenant aléatoirement, un taux d'individus P_c de la population initiale. Similairement pour le nombre de chromosomes qui seront subits à l'opérateur de mutation P_m. Notre approche est basée sur la coopération entre la recherche Tabou et l'algorithme génétique. Dans la partie suivante, nous illustrons notre contribution pour la performance de la recherche Tabou.

5. LA RECHERCHE TABOU

La recherche Tabou est une méta-heuristique basée sur la recherche locale. Elle explore les solutions faisables pour obtenir des solutions optimales ou quasi-optimales. A partir d'une solution initiale, son meilleur voisin est sélectionné à chaque itération. Le processus du voisinage est fait par une simple transition sur la solution courante en utilisant des fonctions de voisinage spéciales. En conséquence, ces solutions sont appelées : solutions voisines. La recherche Tabou utilise une structure de mémorisation dans laquelle, elle sauvegarde la dernière solution visitée (Liste Tabou). Une solution est oubliée pour un certain nombre d'itérations, équivalent à la taille de la liste tabou. Ensuite, le meilleur voisin non tabou sera sélectionné pour participer à la reproduction à la génération suivante de l'algorithme génétique (**Figure 0.1**). La recherche Tabou est une méthode efficace pour plusieurs problèmes, mais son adaptation au problème Job shop Flexible reste difficile. En réalité, elle nécessite la définition de plusieurs paramètres tels que, la solution initiale, les fonctions de voisinage, évaluation des solutions, la taille de la liste tabou, etc.

5.1. Les fonctions de voisinage

Pour le problème d'ordonnancement du job shop flexible (FJSSP), les fonctions de voisinage traditionnelles sont liées à la notion du chemin critique.

Définition 1 : Un chemin critique est un chemin dont la taille égal à la taille du plan (schedule) et il est composé d'opérations liées, soit par des liens de précédence (opérations du même job) et / ou des liens disjonctifs (opérations réalisées par la même machine).

Définition 2 : une opération critique et publique est une opération appartenant à tous les chemins critiques.

Définition 3 : un bloc critique et public est la séquence maximale composée d'opérations adjacentes, critiques et publiques, réalisées par la même machine. N'importe quelle transition est appliquée uniquement sur les blocs critiques et publics.

5.2. Contribution : la fonction de voisinage proposée

Les fonctions de voisinage traditionnelles sont basées sur de simples transitions sur les opérations critiques. Ces transitions peuvent être des permutations entre deux opérations critiques ou une mutation d'une opération critique vers une autre machine. En revanche, ces transitions ne garantissent pas l'amélioration de la solution initiale et parfois, toutes les transitions possibles doivent être explorée afin d'aboutir à une amélioration possible.

Dans ce contexte, nous proposons une fonction de voisinage originale, développée pour trouver des solutions dérivées de meilleure qualité. Au lieu de faire des transitions aléatoires qui peuvent être obsolètes et couteuses de point de vu temps d'évaluation de la nouvelle solution, notre approche est basée sur une analyse profonde de la solution initiale, afin de prendre les décisions, permettant d'apporter une amélioration à chaque opération critique. La fonction de voisinage proposée est basée sur deux algorithmes concurrents qui utilisent des probabilités d'amélioration. Le premier algorithme est pour l'optimisation en utilisant les fenêtres temporelles libres et le deuxième pour l'optimisation à travers les fenêtres temporelles occupées. Les deux algorithmes permettent l'amélioration d'assignement et de séquencement des opérations critiques. Les deux algorithmes sont illustrés dans la partie suivante. Nous avons développé une probabilité pour chaque algorithme afin de mesurer le degré de l'amélioration par chaque algorithme pour l'opération critique en question $O_{j,i,k}$.

L'algorithme ayant la plus grande probabilité, sera sélectionné pour améliorer la date d'achèvement de l'opération critique courante $O_{j,i,k}$.

- Probabilité d'optimisation basée sur les fenêtres temporelles libres (Algorithme A)

$$P_{free-win}(s, O_{j,i,k}) = \frac{1}{1+\left(\frac{PO_{j,i,k}}{Avr\_free\_win_{j,i,k}}\right)} \; et \; (PO_{j,i,k} \leq Avr\_free\_win_{j,i,k}) \qquad \text{Equ. 0-2}$$

- Probabilité d'optimisation basée sur les fenêtres temporelles occupées (Algorithme B)

$$P_{occu-win}(s, O_{j,i,k}) = \frac{1}{1+\left(\frac{PO_{j,i,k}}{Avr\_occu\_win_{j,i,k}}\right)} \; et \; (PO_{j,i,k} \leq Avr\_occ\_win_{j,i,k}) \qquad \text{Equ. 0-3}$$

- S : La solution initiale à optimiser,
- $\text{Avr\_free\_win}_{j,i,k} = \sum_{w=1}^{W_{TF}} (\beta_w - \alpha_w) / WF_{TF}$ la durée moyenne des fenêtres libres pour l'opération $O_{j,i,k}$. W_{tf} est le nombre total de ces fenêtres libres.

- $\text{Avr\_occu\_win}_{j,i,k} = \sum_{w=1}^{W_{TO}}(\text{finish}_w - \text{start}_w)/W_{TO}$ La durée moyenne des fenêtres occupées pour l'opération $O_{j,i,k}$. W_{to} est le nombre total de ces fenêtres occupées.
- $P_{free-win}(s, O_{j,i,k})$: est la probabilité d'améliorer la date d'achèvement de l'opération $O_{j,i,k}$ dans la solution S en exploitant les fenêtres libres (probabilité de l'algorithme A).
- $P_{occu-win}(s, O_{j,i,k})$: est la probabilité d'améliorer la date d'achèvement de l'opération $O_{j,i,k}$ dans la solution S en exploitant les fenêtres occupées (probabilité de l'algorithme B).

Définition d'une fenêtre libre pour $O_{j,i,k}$

$$Max(tfO_{j,i-1} + Tr_{kk'}, \alpha) + P_{j,i,k'} \leq Min(\beta, tO_{j,i+1}, tfO_{j,i,k'}) \quad \text{Equ. 0-4}$$

Définition d'une fenêtre occupée pour $O_{j,i,k}$

$$Max\left(tfO^{-1}{}_{j',i',k'}, tfO_{j,i-1,} + Tr_{kk'}\right) + P_{j,i,k'} \leq Min\left(tO^{+1}{}_{j',i',k'}, tO_{j,i+1}\right)$$

$$Max\left(tfO^{-1}{}_{j,i,k}, tfO_{j',i'-1} + Tr_{k'k}\right) + P_{j',i',k} \leq Min(tO^{+1}{}_{j,i,k}, tO_{j',i'+1}) \quad \text{Equ. 0-5}$$

L'exploitation d'une fenêtre occupée pour améliorer une opération critique $O_{j,i,k}$ se traduit par une permutation entre cette opération et celle qui occupe la fenêtre. En revanche, les deux formules 3.6 assurent le respect des contraintes afin de ne pas retarder les autres opérations.

5.2.1. L'algorithme a pour l'optimisation basée sur les fenêtres libres

Algorithm A

Input: initial schedule (S), $O_{j,i,k}$
Output: optimized schedule (S')
Begin
$Tab_{\lambda F} \longleftarrow$ Get_Free_Windows(S, $O_{j,i,k}$) //avoir la première fenêtre libre dans Mk
$tf_{Best} \longleftarrow tfO_{j,i,k}$ //meilleur temps de fin initial de $O_{j,i,k}$
$M_{Best} \longleftarrow M_k$ //la meilleure machine pour $O_{j,i,k}$
For w = 1 **to** W_F **do** //explorer les fenêtres libres ordonnées W_F
$\lambda^{k'}{}_W = Tab_{\lambda F}[w]$ //obtenir la fenêtre libre actuelle de $M_{k'}$
$(\alpha, \beta) \longleftarrow$ Bornes($\lambda^{k'}{}_W$) //les bornes de la fenêtre libre
If Max($tfO_{j,i-1} + Tr_{kk'}, \alpha$) + $P_{j,i,k'} \leq$ Min($\beta, tO_{j,i+1,k}, tfO_{j,i,k}$) **then** //déplacer $O_{j,i}$ vers $M_{k'}$
If Max($tfO_{j,i-1} + Tr_{kk'}, \alpha$)+ $P_{j,i,k'} < tf_{best}$ then
$M_{Best} \longleftarrow M_{k'}$; $tO_{j,i} \longleftarrow$ Max($tfO_{j,i-1} + Tr_{kk'}, \alpha$); $tf_{Best} \longleftarrow tO_{j,i} + P_{j,i,k'}$
end$_{if}$
end$_{if}$
end$_{for}$
if ($tf_{Best} \neq tf_{j,i,k}$ or $P_{j,i,best} < P_{j,i,k}$) **then**

update_schedule(S, $tO_{j,i}$) //démarrer la mise à jour
end$_{if}$
End.

L'algorithme A peut améliorer l'assignement et le séquencement d'une opération critique par la détection des meilleure fenêtres temporelles libres adéquates à cette opération critique. Ceci est du sans briser les contraintes de ressources et précédence ou retarder d'autres opération.

Les fenêtres libres sont détectées à partir de toutes les machines candidates pour l'opération critique courante. Ensuite, on vérifie si l'amélioration courante est la meilleure parmi les améliorations précédentes. A la fin de l'algorithme, nous démarrons la procédure de mise à jour afin de modifier les dates de début et de fin des opérations concernées.

5.2.2. L'algorithme b pour l'optimisation basee sur les fenetres occupees

L'algorithme B commence par la détection des fenêtres occupées adéquates à l'opération critique courante $O_{j,i,k}$. l'opération qui occupe la fenêtre occupée courante est appelée Oj',i',k'. L'idée de cet algorithme est d'échanger les position des deux opérations $O_{j,i,k}$ et Oj',i',k' afin d'améliorer l'assignement et/ou le séquencement de l'opération critique $O_{j,i,k}$ tout en respectant les contraintes du problème pour les deux opérations $O_{j,i,k}$ et Oj',i',k'.

L'algorithme B choisi explore les fenêtres occupées adéquates à l'opération $O_{j,i,k}$ et sélectionne celle qui améliore la date d'achèvement de $O_{j,i,k}$.

Algorithm B
Input: initial schedule (S), $O_{j,i,k}$
Output: optimized schedule (S')
Begin
$Tab_{\lambda O} \longleftarrow$ Get_Occupied_Windows(S, $O_{j,i,k}$) //avoir la première fenêtre occupée dans Mk
$tf_{Best} \longleftarrow tfO_{j,i,k}$ //meilleur temps de fin initial de$O_{j,i,k}$
$M_{Best} \longleftarrow M_k$ //meilleure machine pour $O_{j,i,k}$
For w = 1 **to** W_O **do** //explorer les fenêtres occupées ordonnées W_O
$O_{j',i',k'} \longleftarrow get\_oper(Tab_{\lambda O}[w])$ //avoir l'opération occupant la fenêtre
If Max($tfO^{-1}_{j',i',k'}, tfO_{j,i-1} + P_{j,i,k'} + Tr_{kk'}$) $\leq Min\ (tO^{+1}_{j',i',k'}, tO_{j,i+1},\ tfO_{j,i,k})$ **then**
//déplacer $O_{j,i}$ vers $M_{k'}$
If Max($tfO^{-1}_{j,i,k}, tfO_{j',i'-1} + P_{j',i',k} + Tr_{k'k}$) $\leq Min\ (tO^{+1}_{j,i,k}, tO_{j',i'+1})$ **then**
//déplacer $O_{j',i'}$ vers M_k
If Max($tfO^{-1}_{j',i',k'}, tfO_{j,i-1} + P_{j,i,k'} + Tr_{kk'}$) $< tf_{Best}$ **then** //déplacer $O_{j',i'}$ vers M_k
$M_{Best} \longleftarrow M_{k'}$; $tO_{j,i} \longleftarrow$ Max ($tfO^{-1}_{j',i',k'}, tfO_{j,i-1} + Tr_{kk'}$) //temps du début
$tf_{Best} \longleftarrow tO_{j,i} + P_{j,i,k'}$; $tO_{j',i'} \longleftarrow$ Max ($tfO^{-1}_{j,i,k}, tfO_{j',i'-1} + Tr_{k'k}$)
end$_{if}$
end$_{if}$
end$_{if}$
end$_{for}$
if ($tf_{Best} \neq tf_{j,i,k}$ *or* $P_{j,i,best} < P_{j,i,k}$) **then** //démarrer la mise à jour
update_schedule(S, $Min(tO_{j,i}, t_{j',i',k})$)
end$_{if}$

End.

Apres avoir optimisé l'opération critique $O_{j,i,k,}$ Les deux algorithmes A et B utilisent une procédure de mise à jour. Le pseudo algorithme de la procédure de mise à jour est présenté comme suit :

Update_schedule

Input: initial schedule (S), t_{min}
Output: new schedule (S')
Begin
  For o = 1 **to** o_T **do** //explorer toutes les opérations
    $O_{j\ ,i\ ,k}$ ⟷ Tab_O[o] //obtenir l'opération en cours
    If $tO_{j\ ,i\ ,k} \geq t_{min}$ **then**
      $tO_{j\ ,i\ ,k}$ ⟷ Max ($tfO_{j\ ,i\ -1,k'} + Tr_{k'k}, tfO^{-1}{}_{j\ ,i\ ,k}$) //déplacement du k' vers k
      $tfO_{j\ ,i\ ,k}$ ⟷ $tO_{j\ ,i\ ,k} + P_{j\ ,i\ ,k}$
    end$_{if}$
  end$_{for}$
End.

Seulement les opérations qui se trouvent après la date t_{min} qui seront mises à jour.

Les deux algorithmes précédents jouent le rôle de la fonction de voisinage pour la recherche Tabou (TS). Afin d'augmenter l'efficacité de la recherche Tabou, nous avons proposé trois critères pour sélectionner les individus qui seront soumis à TS en utilisant la fonction fitness. Les trois critères proposés pour la sélection des individus qui seront soumis à TS sont les suivants :

- *Makespan* : ce critère indique la qualité de l'ordonnancement. Il représente le temps d'achèvement de tous les jobs (Equ. 0-1).
- *La durée moyenne des fenêtres libres* : le temps moyen inexploité renforce l'efficacité de l'algorithme A.

$$Avr_{WF} = \sum_{k=1}^{M} \frac{WF_k}{M} \qquad \text{Equ. 0-6}$$

Où WF_k est la durée totale du temps inexploité par la machine M_k.

- *La distribution de la charge de travail entre les machines* : un plan plus équilibré facilité l'application de l'algorithme B (optimisation par les fenêtres occupées).

$$\delta_W = \sum_{k=1}^{M} \frac{W_{k*} - W_k}{W_T} \qquad \text{Equ. 0-7}$$

$W_{k*} = MAX[W_k], k \in [1, M]$: la plus grande charge de travail parmi toutes les machines.

W_k : la charge de travail de la machine M_k.

W_T : La charge de travail de toutes les machines.

En se basant sur ces trois critères, les chromosomes enfants qui seront soumis à la recherche Tabou, seront sélectionnés par la formule suivante :

$$Fitness\_TS(S) = \frac{AVR_{W_F}}{C_{max} \ * \ (\delta_W + 1)} \qquad \text{Equ. 0-8}$$

Le but de la formule 3.8 est de sélectionner les chromosomes qui ont une durée moyenne du temps inexploité (fenêtres libres) maximale, un makespan minimum et une distribution équilibrée de la charge de travail entre les machines.

5.2.3. L'algorithme de la recherche tabou (TS)

Le pseudo algorithme résume comment la recherche Tabou améliorée a été implémentée (ITS : Improved Taboo Search).

Improved _Taboo_Search

Input: initial solution (S)
Output: the best neighbor (S_n)
Begin
$taboo_{list} \leftarrow \emptyset$; $solution_{current} \leftarrow solution_{initial}$;
$solution_{best} \leftarrow solution_{current}$; $nb_{iterations} \leftarrow 0$;
TabCriticalOperation //initialiser la table des opérations critiques
while ($nb_{iterations} \leq nbMax_{iterations}$) **do**
for o = 1 to $nb_{critical\ \_operations}$ **do** //explorer les opérations critiques
$O_{j,i,k} \leftarrow$ TabCriticalOperation[o] //avoir l'opération critique en cours de TabCriticalOperation}
//utiliser les algorithmes A, B pour le meilleur voisinage
$neighbour_{bestj\ ,i,k} \leftarrow getBestNeighbour(Alg(A), Alg(B), O_{j,i,k}, S)$
if $(cost(neighbour_{best}) < cost(solution_{current}))$**then** {makespan}
If ($solution_{current} \notin taboo_{list}$)
$solution_{best} \leftarrow neighbour_{best}$; $nb_{iteration}$ ++
$taboo_{list} \leftarrow addTaboo(neighbour_{bestj\ ,i,k})$
end$_{if}$
end$_{for}$
end$_{while}$
End.

L'idée générale de la recherche tabou, consiste à explorer le voisinage d'une solution initiale, afin de trouver la meilleure solution de ce voisinage. L'algorithme commence par initialiser la liste des solutions taboues ainsi que la solution courante. Ensuite, la procédure TabCriticalOperation détecte les opérations critiques (dont le temps d'achèvement est équivalent au makespan). Apres, un processus d'optimisation à travers les deux algorithmes A, B sera déclenché pour chaque opération critique. L'application des algorithmes A, B conduit à améliorer le temps d'achèvement de l'opération critique, ce qui contribue à réduire le makespan.

6. L'ALGORITHME GENETIQUE

La recherche Tabou présentée dans la section précédente sera utilisée pour améliorer les chromosomes enfants générés par l'opérateur de croisement. La recherche tabou proposée a un double rôle : elle aide à la découverte de nouvelles solutions et contribue à accélérer la vitesse de converge de l'algorithme génétique en réduisant le nombre d'itérations nécessaires afin de trouer une solution de bonne qualité.

L'algorithme génétique est l'une des meilleures métas heuristiques pour la résolution des problèmes d'optimisation combinatoires. Cet algorithme a un opérateur de croisement pour la recombinaison des gènes parentaux afin d'engendrer des chromosomes enfants. Le deuxième opérateur génétique est la mutation, utilisé afin de changer aléatoirement les valeurs de quelques gènes, afin d'éviter une convergence prématurée de l'algorithme génétique. Mais, vu la nature combinatoire du problème considéré, trouver une solution optimale ou quasi optimale reste une tâche difficile à atteindre. Dans ce contexte, nous proposons de combiner la recherche Tabou avec l'algorithme génétique pour obtenir des solutions de meilleure qualité (**Figure 0.1**).

L'algorithme génétique améliore à travers plusieurs générations la population initiale. En revanche, la qualité de cette population initiale est très importante pour la performance de l'algorithme génétique. Une population initiale de bonne qualité doit être composée d'individus diversifiés et de bonne qualité. Dans la sous section suivante, nous illustrons la génération de la population initiale, dans le but d'assurer la diversité et la qualité des solutions.

6.1. La population initiale

Il existe plusieurs méthodes qui permettent de générer la population initiale. Mais, généralement cette population est générée aléatoirement, la chose qui peut empêcher l'algorithme génétique à explorer profondément l'espace de solution. Dans notre cas, la génération de la population initiale se fait de la manière suivante :

- 50 % de la population est générée aléatoirement afin d'assurer la diversité, ce qui est utile afin d'éviter une convergence prématurée.

- le reste de la population est généré par des heuristiques telles que SPT (shortest processing time), MWR (most work remaining), etc.

Dans cette partie, nous proposons une heuristique (ASH : Active Schedule Heuristic) capable de trouver la meilleure fenêtre libre pour chaque opération. Cette heuristique génère un plan actif, dans lequel, nous ne pouvons pas avancer une telle opération sans retarder une autre.

- $W_k = \sum_{j=1}^{n} \sum_{i=1}^{o_{nj}} P(O_{j,i,k}) * X_{j,i,k}$: la charge de travail de la machine M_k

- $X_{j,i,k} = 1$ if $O_{j,i}$ est assignée à M_k, else $X_{j,i,k} = 0$
- $W_T = \sum_{k=1}^{M} W_k$: la charge de travail totale
- $WR_k = \frac{W_k}{W_T}$: le taux de la charge de travail de la machine M_k
- $S_{k,X}$: La date de début de la fenêtre libre X situées à la machine M_k. La formule 3.10 mesure le cout de placement de l'opération $O_{j,i}$ sur la machine M_k et précisément dans la fenêtre libre X.

$$Cost(O_{j,i,k})_X = MAX\left(tf[O_{j,i-1}] + Tr_{k'k}, S_{k,X}\right) + P(O_{j,i,k}) + WR_k \quad \text{Equ. 0-9}$$

$$best\_window_k = MIN\left(Cost(O_{j,i,k})_X, .., Cost(O_{j,i,k})_{nb_x}\right) \quad \text{Equ. 0-10}$$

La formule 3.10 détecte la meilleure fenêtre libre pour l'opération $O_{j,i}$ parmi l'ensemble nb_x des fenêtres situées à la même machine Mk. Cette formule proposée nous permet de trouver la meilleure fenêtre libre parmi toutes les fenêtres disponible dans toutes les machines candidates.

6.2. Codage de chromosome

Afin d'exploiter l'algorithme génétique, chaque solution doit être codée selon une représentation spécifique pour obtenir un chromosome (**Table 0.2**). Cette représentation sera utilisée afin d'explorer de nouvelles solutions. Cette représentation est simple et linéaire.

MACHINE	2	1	3	3	...
OPERATION	$O_{3,1}$	$O_{1,1}$	$O_{3,2}$	$O_{3,3}$	...

Table 0.2 Représentation d'un chromosome

Assurer les contraintes de précédence des jobs est nécessaire. En revanche, éviter un processus de correction sur les chromosomes générés, qui incrémente dramatiquement l temps de calcul, sera très important. C'est pour cette raison que nous avons utilisé l'opérateur de croisement POX (Lee, Yamakawa et Lee 1998), qui ne nécessite pas la correction des chromosomes générés.

Dans le deuxième opérateur génétique, la mutation est appliquée en choisissant aléatoirement un chromosome et une machine, sur laquelle la perturbation sera appliquée. Ensuite, nous choisissons aléatoirement deux opérations appartenant à deux jobs différents et nous les permutons.

6.3. Evaluation des chromosomes

Dans notre cas, nous avons adopté le critère de Cmax et représente la durée de production de tous les jobs. Il reflète la qualité d l'ordonnancement. La fitness des chromosomes est calculée par la formule suivante :

$$fitness(chromosome) = 1/Cmax \qquad \text{Equ. 0-11}$$

6.4. Sélection

La sélection des chromosomes parents est cruciale pour choisir les individus qui participeront au processus de reproduction pour la génération suivante. Deux individus sont aléatoirement sélectionnés et celui ayant la fitness la plus importante qui sera sélectionné comme parent. Cette procédure est répétée jusqu'à ce que tous les parents soient sélectionnés.

6.5. Croisement

Cet opérateur échange les informations des parents sélectionnés afin d'obtenir des enfants de meilleure qualité. Le croisement est appliqué sur chaque paire de chromosomes (parent) pour obtenir deux nouveaux individus (enfants).

Nous avons adopté une technique de croisement appelée improved Precedence Preserving Order-based crossover (POX) (Lee, Yamakawa et Lee 1998). Les étapes de POX sont présentées comme suit :

a) Générer deux sous ensembles de jobs JS1-JS2 à partir de tous les jobs et sélectionner deux chromosomes parent OS1 - OS2 aléatoirement.
b) Copier chaque élément de OS1 – OS2 qui appartient à JS1 – JS2 dans les chromosomes enfants OS1' – OS2' et garder les mêmes positions.
c) Supprimer les éléments qui existent déjà dans les sous ensembles JS1 – JS2 de OS1– OS2.
d) Remplir les gènes vides dans OS1' – OS2' des éléments de OS1 – OS2 en respectant le même ordre.

En se basant sur l'exemple donné à la **Table 0.1**, la figure suivante (**Figure 0.2**) illustre le principe de l'opérateur génétique POX avec 5 jobs.

Les deux sous ensembles des jobs JS1 = {1,3} et JS2 = {2,4,5}. Ensuite, chaque élément dans OS1 qui appartient à JS1 sera copié à OS1'dans la même position et au même temps, supprimer chaque élément qui appartient à JS1 de OS2.

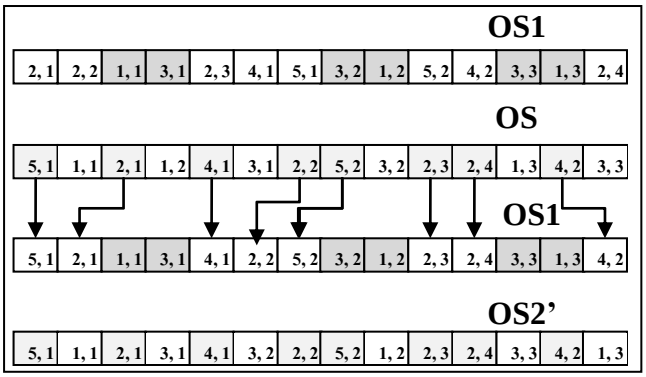


Figure 0.2 Exemple de l'opérateur de croisement POX

6.6. Mutation

Afin de maintenir la diversité de la population, la mutation est appliquée par altération de quelques gènes. Cette opération est appliquée selon une petite probabilité, parce qu'une mutation intense pourrait casser les chromosomes. L'application de cet opérateur est divisée en deux parties :

6.6.1. Mutation d'assignement

Au départ, nous sélectionnons deux indexes différent L1 et L2 appartenant à l'intervalle {1,Nbo}, dont Nbo et le nombre total d'opérations de tous les jobs. Ensuite, nous choisissons aléatoirement pour chaque gène entre L1 et L2 une autre machine à partir des machines candidates (**Figure 0.3**).

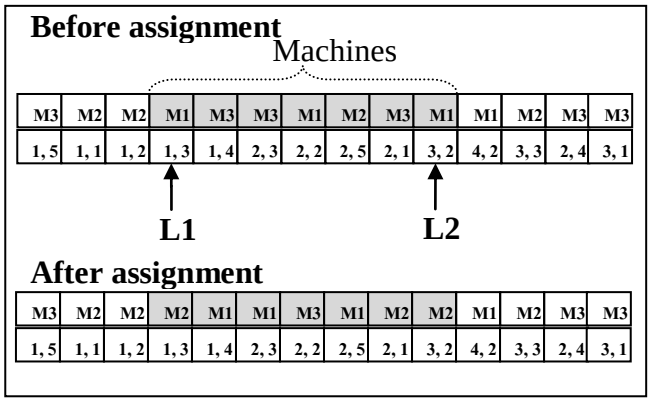


Figure 0.3 Exemple de mutation d'assignement

6.6.2. Mutation de séquencement

Nous sélectionnons aléatoirement deux opérations différentes (de différents jobs) ensuite, nous permutons ces deux opérations (**Figure 0.4**).

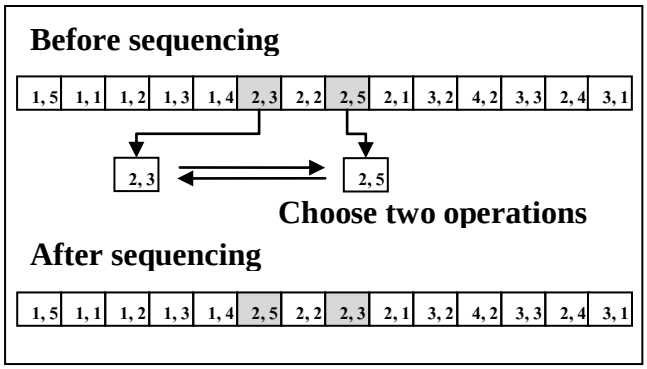


Figure 0.4 Exemple de mutation de séquencement

6.7. Remplacement et élitisme

Afin de garantir que l'exploration ne diverge pas vers une solution ayant une valeur de fitness importante, toutes les solutions à chaque itération sont remplacées, sauf la meilleure (la solution élite) (Sandhya, Kumar et Satapathy 2013).

6.8. Les paramètres de l'algorithme génétique

La performance de l'algorithme génétique dépend fortement d'un ensemble de paramètres. Apres plusieurs tests, l'algorithme génétique est configuré comme suit :

La probabilité de croisement Pc doit être supérieur à 0.6 afin de bien explorer l'espace de solutions et de trouver de nouveaux descendants. Par contre, la probabilité de mutation Pm est entre 0.01 et 0.02, parce que, le but de mutation est d'appliquer une légère perturbation sur chaque chromosome pour la diversification de la population et d'éviter une convergence prématurée de l'algorithme génétique. Une probabilité de mutation Pm importante peut détruire le chromosome.

7. CAS D'ETUDE : AIP-PRMECA

Les benchmarks sont utilisés pour comparer les performances des systèmes. Plusieurs Benchmarks sont proposés dans la littérature de la recherche opérationnelle. (Beasley 1990) , (Taillard 1993) et précisément (Brandimarte 1993) pour le problème d'ordonnancement du job shop flexible.

Mais, ces Benchmarks sont théoriques et ne satisfassent pas les contraintes des systèmes manufacturiers réels. Nous avons la chance d'utiliser un benchmark spécial, développé au sein de l'université de Valencienne, France (Trentesaux, et al. 2013). Ce benchmark est basé

sur un cas d'étude réel : AIP-PRIMECA (**Figure 0.5**). En plus, le temps de transport entre les machines, qui est important dans le monde réel, est pris en considération.

Ce Benchmark offre la possibilité d'étudier des scénarios de perturbations, tels que les pannes des machines, les ordres d'urgence, apparition d'une nouvelle ressource, etc. Ces scénarios permettent de tester la robustesse d'une méthode de pilotage.

Dans la partie suivante, les données de la cellule de production AIP-PRIMECA seront présentées. Nous commençons par les données liées aux différents produits, ensuite celle relatives aux ressources et les données du système de transport (Trentesaux, et al. 2013).

Figure 0.5 Cas d'étude, AIP-PRIMECA

7.1. Les données sur les produits

La cellule de production AIP-PRIMECA dispose de cinq composants de base (Figure 3.6 a) : « Axe_comp », « I_comp », « L_comp », « r_comp » et « Vis_comp ». En plus, un plateau accueillant l'assemblage. L'assemblage de ces composants se fait sur un plateau vierge, déposé sur chaque navette (ressource de transport, Figure 3.6 c). Ces composants sont transportés d'une machine à une autre pour l'assemblage et retournent à la première ressource pour décharger le produit fini. Chaque ressource d'assemblage dispose d'un stock de pièces. Nous supposons que le stock est illimité.

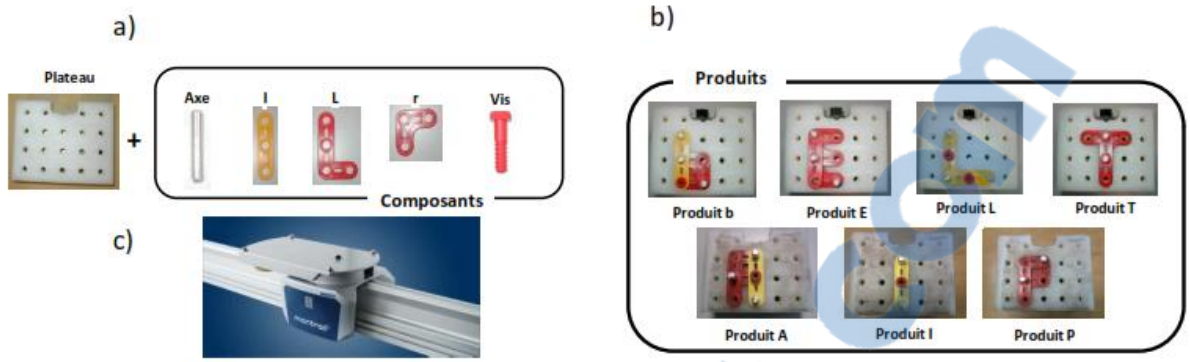


Figure 0.6 Composants, Produits et navette

En utilisant les cinq composants cités précédemment, il est possible d'assembler sept produits différents : « B », « E », « L », « T », « A », « I », « P » (**Figure 0.6** b). La fabrication de chaque produit commence toujours par une opération de chargement du plateau et le déchargement du produit à la fin.

Chaque produit a une gamme opératoire, composée d'un ensemble d'opérations élémentaires, assurées par des ressources de la cellule. Il existe huit opérations élémentaires : « Chargement_plateau », « Montage_Axe », « Montage_I », « Montage_L », « Montage_r », « Montage_Vis », « Inspection automatisée » et « Déchargement_plateau ». À chaque fois qu'une ressource reçoit un plateau, elle récupère la pièce adéquate de son stock, en fonction du produit désiré et commence l'assemblage. Avant qu'un produit quitte la cellule, il passe par une station d'inspection basée sur un système de vision.

La fabrication d'un tel produit passe par une séquence de production spécifique (**Table 0.3**). Toutes les séquences ont la même structure : Chargement du plateau vierge sur la navette, une série d'opérations d'assemblage, l'inspection du produit pour détecter les défauts et à la fin, le déchargement du produit fini.

Dans le cas ou deux opérations consécutives appartenant à la même séquence de production sont effectuées par deux machines différentes, un temps de transport est nécessaire.

B	E	L	T	A	I	P
Load	Load	Load	Load	Load	Load	Load
Axis	Axis	Axis	Axis	Axis	Axis	Axis
Axis	Axis	Axis	Axis	Axis	Axis	Axis
Axis	Axis	Axis	Axis	Axis	Axis	Axis
r_comp	R_comp	I_comp	L_comp	R_comp	Screw	L_comp
r_comp	R_comp	I_comp	Inspection	L_comp	Inspection	Inspection
I_comp	L_comp	Screw	Unload	I_comp	Unload	Unload
Screw	Inspection	Screw		Screw		
Inspection	Unload	Inspection		Inspection		
Unload		Unload		Unload		

Table 0.3 Les séquences de production

Trois ordres de fabrication sont réalisables : « BELT », « AIP » et « LATE ». Ces ordres de fabrication des clients comportant de trois jusqu'à quatre produits, peuvent être fabriqués dans n'importe quel ordre. Une fois que tous les produits sont réalisés, l'ordre de fabrication sera achevé. La **Figure 0.7** illustre présente les différents ordres de fabrication.

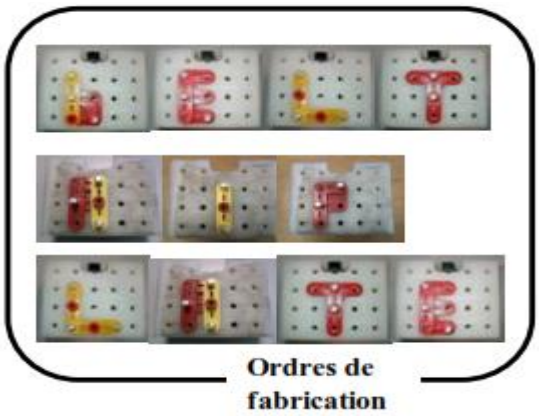


Figure 0.7 Les ordres de fabrication

7.2. Les données sur les ressources

Sept ressources sont intégrées dans la cellule de production AIP-PRIMCA (Figure 3.8):

- Unité de chargement / déchargement (Machine 1)
- Trois stations d'assemblage (Machine 2, Machine 3, Machine 4)
- Unité d'inspection automatique (Machine 5)
- Unité de réparation, qui est la seule unité manuelle.
- Une station optionnelle, utilisée dans certain scenarios de perturbation.

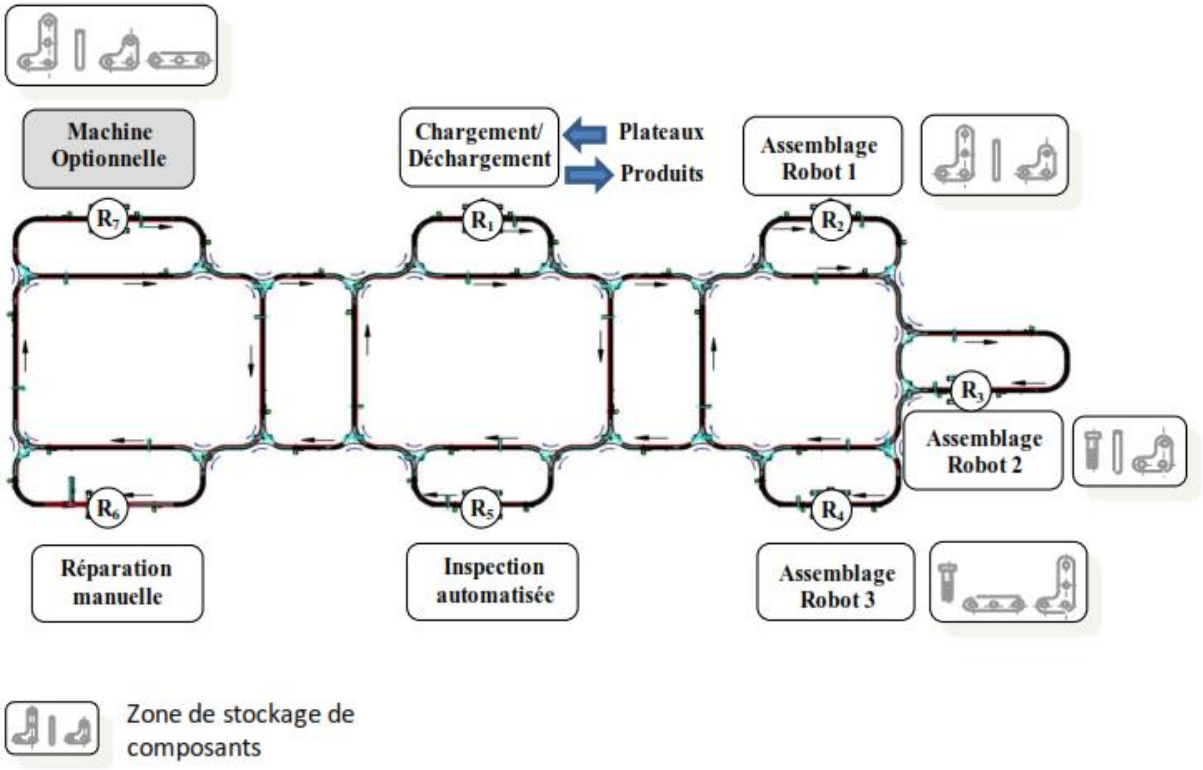


Figure 0.8 Localisation des ressources

Certaines machines (ressources) sont capables d'assurer plusieurs opérations élémentaires, par contre, certaine opérations sont assurée par une seule machine, comme le chargement et le déchargement des plateaux. Chaque station d'assemblage a sa propre zone de stockage, utilisée pour alimenter la machine par les pièces nécessaires en fonction du produit désiré.

La **Table 0.4** présente les opérations élémentaires assurées par chaque machine. Les valeurs indiquent le temps opératoire nécessaire pour achever telle opération par telle machine. Chaque machine ne peut réaliser qu'une seule opération à la fois et l'exécution d'une opération ne peut pas être interrompue. La machine 7 n'est pas utilisée en mode de fonctionnement normal (sans perturbation).

Id	Operations	M1	M2	M3	M4	M5	M6	M7
#1	Axis		20	20				(20)
#2	r_com		20	20				(20)
#3	I_comp				20			(20)
#4	L_comp		20		20			(20)
#5	Screw_comp			20	20			
#6	Inspection					5		
#7	Loading	10						
#8	Unloading	10						
#9	Recovering						60	

Table 0.4 Les durées opératoires des opérations élémentaires

7.3. Les données sur le système de transport

Les machines sont connectées par un système de transport. Ce système est un monorail unidirectionnel avec des aiguillages rotatifs avant chaque divergence / convergence. Ainsi, ce système peut être considéré comme un graphe fortement connexe, composé des nœuds suivants (**Figure 0.9**) :

- R1, R2, R3, R4, R5, R6, R7 sont des nœuds de ressources (machines),
- n1, n2, n3, n4, n5, n6, n7, n8, n9, n10, n11 sont des nœuds décisionnels de routage.

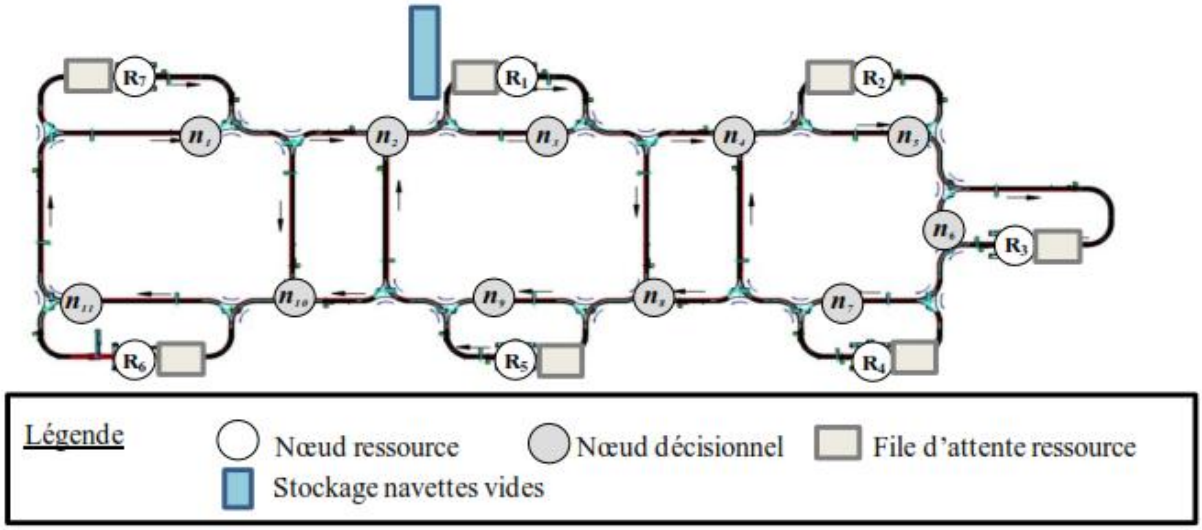


Figure 0.9 Système de transport

Les navettes sont autopropulsées et peuvent transporter un produit de nœud en nœud. Le nombre maximum de navettes disponibles est 10. Donc, à chaque instant, 10 produits (jobs) peuvent être manufacturés au même temps. Les navettes sont dotées d'un mécanisme permettant de gérer la vitesse afin d'éviter la collision avec d'autres navettes. La première navette qui entre sur un tronçon sera toujours la première à en sortir (First In First Out).

Pour la simplification, Les navettes vides sont supposées êtres rangées dans une zone à côté de la machine 1 et dont la capacité de stockage des navettes est illimitée. Un mécanisme extérieure est considéré qui charge et décharge les navettes à cet endroit. Donc, c'est l'endroit de départ d'arrivée des navettes. Elles débutent par la récupération d'un plateau (chargement au niveau de M1) afin d'entrer dans le système. Elles transportent les produits d'une machine à une autre, afin d'achever l'assemblage et reviennent au point de départ pour décharger le produit au niveau de la machine 1. Il est impossible qu'une navette vide circule dans la cellule.

Les temps de transport entre les nœuds de la cellule sont fixés dans la **Table 0.5**.

	n1	n2	n3	n4	n5	n6	n7	n8	n9	n10	n11	m1	m2	m3	m4	m5	m6	m7
n1		4								5								
n2			4									5						
n3				4				5										
n4					4								5					
n5						4								11				
n6							4								5			
n7				5				4										
n8									4									
n9		5								4								
n10											4						7	
n11	9																	10
m1				6				7										
m2						5								13				
m3							6								7			
m4				7				6										
m5		7								6								
m6	12																	13
m7		6								7								

Table 0.5 Les temps de transport

8. EXPERIMENTATIONS ET RESULTATS

L'approche d'ordonnancement proposée a été implémentée en utilisant le langage de programmation JAVA. L'éditeur utilisé est ECLIPSE. Nous considérons un scénario statique. Les perturbations ne sont pas prises en compte, contrairement aux scénarios dynamiques. En revanche, le scénario statique peut se préparer en mode off-line permettant de développer des mécanismes d'optimisation avant de passer à la simulation réelle.

Dans cette expérience, nous évaluons la qualité de l'individu de la population initial, obtenu en utilisant la méthode de génération aléatoire de la population initial et l'heuristique proposée (ASH : Active Schedule Heuristic). Les résultats obtenus sont présenté dans la **Table 0.6**.

Product order	**Random**	**ASH***
1 x A-I-P	310	**260**
2 x A-I-P	460	**388**
3 x A-I-P	589	**480**
B-E-L-T-A-I-P	601	**496**

Table 0.6 Comparaison entre la méthode aléatoire et ASH pour la population initiale

Les résultats obtenus montrent que la qualité moyenne de la population initiale a été améliorée par 18%, en comparaison avec la génération aléatoire de la population initiale. Une comparaison est établie entre l'approche proposée IGA (Improved Genetic Algorithm) et cinq autres approches :

Une méthode exacte MILP(Mixed-Integer Linear Program), PFA(Potential Field Approach), IGIT(Iterative Greedy Insertion Technique), (GA Genetic Algorithm) et ORCA-FMS.

- *The Mixed-Integer Linear Program*

MILP présente le modèle formel du benchmark, qui a été implémenté sous CPLEX (Trentesaux, et al. 2013).
MILP offre une solution off-line qui respecte toutes les contraintes. Le nombre de variables entières ainsi que le nombre de contraintes détermine la complexité de la méthode.

- *The potential fields*

Cette méthode est largement utilisée pour le cheminement des robots mobiles dans les environnements statiques et dynamiques. Elle est basée sur des champs potentiels artificiels, qui attirent les robots vers le but et les repoussent des obstacles (Raja et Pugazhenthi 2011). Elle a été utilisée pour la première fois pour piloter des robots mobiles (Khatib 1986).

- 3 *The Iterative Greedy Insertion Technique*

Cette technique est très populaire pour résoudre les problèmes d'optimisation combinatoires (Talb 2009). Elle fonctionne d'une manière récursive et destructive. Chaque itération écrase sa précédente, jusqu'à ce que l'optimum local soit trouvé (Bekkar, et al. 2016).

- ORCA-FMS

Cette approche proposée récemment est basée sur une architecture holonique (C. Pach, et al. 2014). Des entités autonomes présentant les produits et les ressource sont utilisées pour la prise de décision. En revanche, cette architecture n'est pas purement décentralisée, par ce que ce modèle est guidé par une méthode exacte ILP (Integer Linear Programming).

La **Table 0.7** Présente le résultat de cette comparaison.

Compagne de production	**MILP**	**PF**	**IGIT**	**GA**	**ORCA-FMS**	**IGA***
1 x A-I-P	219*	-	229	224	224	**219**
2 x A-I-P	309*	-	346	335	314	**313**
3 x A-I-P	409**	-	522	425	415	416
B-E-L-T-A-I-P	570**	448	431	410	-	**396**

Table 0.7 Résultats obtenus de cmax en secondes
* Solution optimale
** Meilleure solution trouvée

Les résultats présentés dans la **Table 0.7** représentent une comparaison entre notre approche IGA (Improved Genetic Algorithm) et cinq autres méthodes : MILP(Mixed Integer

Linear Program), PF(Potential Field), IGIT(Iterative Greedy Insertion Technique) et AG (Basic Genetic Algorithm).

D'après les résultats obtenus, pour la première compagne de production 1 x A-I-P, une solution optimale est obtenue par IGA. La même solution a été trouvée par la méthode exacte MILP. Il y a une amélioration de 4.5% par rapport à IGIT et 2.3% par rapport à GA et ORCA-FMS.

Pour la deuxième compagne de production 2 x A-I-P, IGA offre la deuxième meilleure solution, après la méthode exacte MILP (pas loin de la solution optimale), avec une amélioration de 9.6% par rapport à IGIT et 6.6%, 0.4 % par rapport à GA et ORCA-FMS.

La meilleure solution pour la troisième compagne de production 3 x A-I-P, a été obtenue par la méthode exacte MILP. Ensuite, Les deux meilleures solutions ont été trouvées par ORCA-FMS et IGA avec une différence de 0.2% pour ARCA-FMS.

La quatrième compagne de production B-E-L-T-A-I-P n'a pas été testée sur ORCA-FMS. La meilleure solution a été trouvé par IGA avec une amélioration de 30.6% par rapport à MILP et 9.2%, 8.2% et 3.5% par rapport à PF, IGIT et GA.

D'après les résultats obtenus, nous constatons que les solutions obtenues par la méthode proposée sont de bonne qualité. En plus, il est même possible d'atteindre l'optimalité, comme le cas pour le premier ordre de production 1xA-I-P.

Nous présentons dans les figures suivantes : **Figure 0.10**, **Figure 0.11**, **Figure 0.12** et **Figure 0.13** les solutions obtenues par IGA :

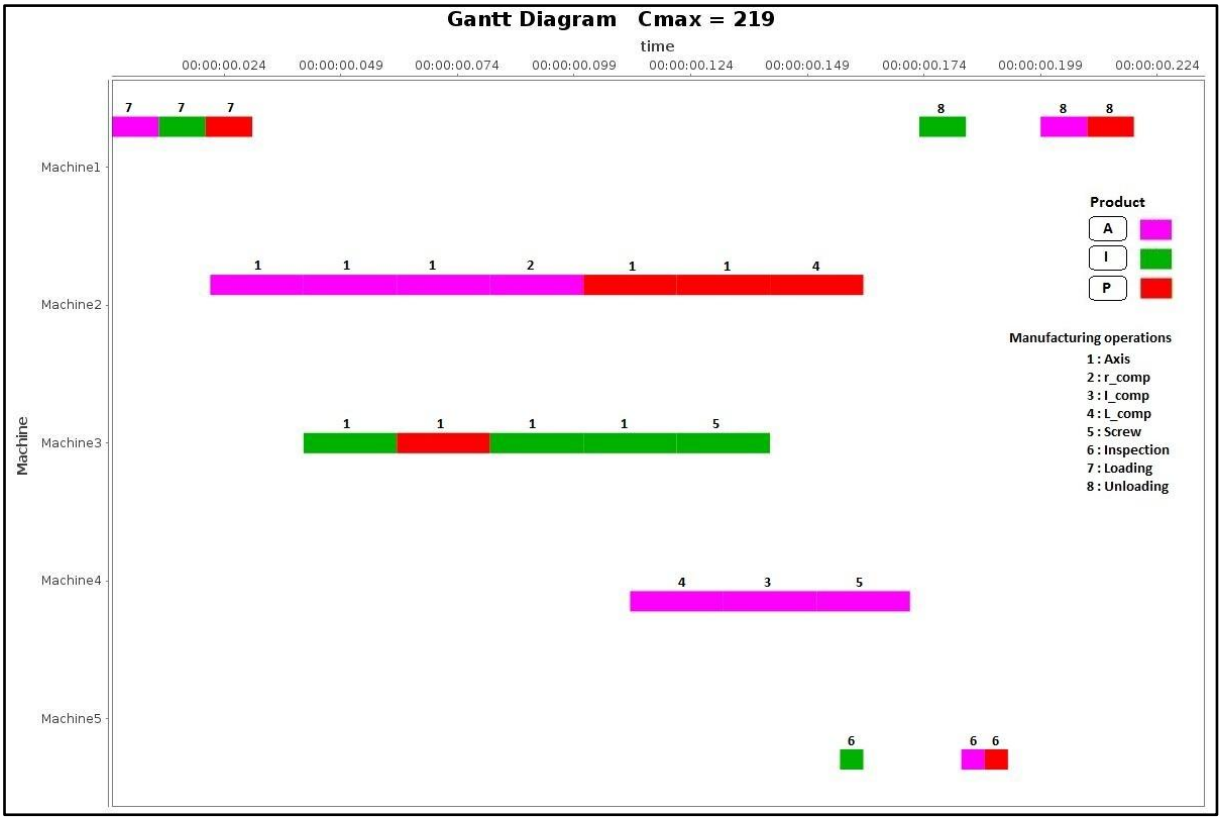


Figure 0.10 Diagramme de Gantt pour 1xA-I-P (IGA)

La machine 1 charge un nouveau plateau pour chaque produit (job) qui sera transporté vers les autres machines pour l'assemblage du produit. Quand toutes les opérations nécessaires sont réalisées, la machine 1 décharge le plateau du produit fini dans la zone de chargement/déchargement. Tous les produits passent par l'inspection faite par la machine 5.

La **Figure 0.10** montre que le temps durant lequel les machines attendent l'arrivée des produits est limité. Pour la machine 1, après avoir chargé les plateaux, cette machine n'intervient qu'à la fin de la production pour décharger les produits finis. Nous constatons une seule fenêtre temporelle, dans laquelle la machine 1 est inexploitée. La machine 2 attend l'arrivée du premier produit après une opération de chargement par la machine 1. Il n'existe aucune fenêtre où la machine 2 est inexploitée. Les machines 3 et 4 sont parfaitement exploitées. La machine 5 responsable de l'inspection des produits a une seule fenêtre où cette machine est inexploitée. La solution présentée dans la **Figure 0.10** est une solution optimale.

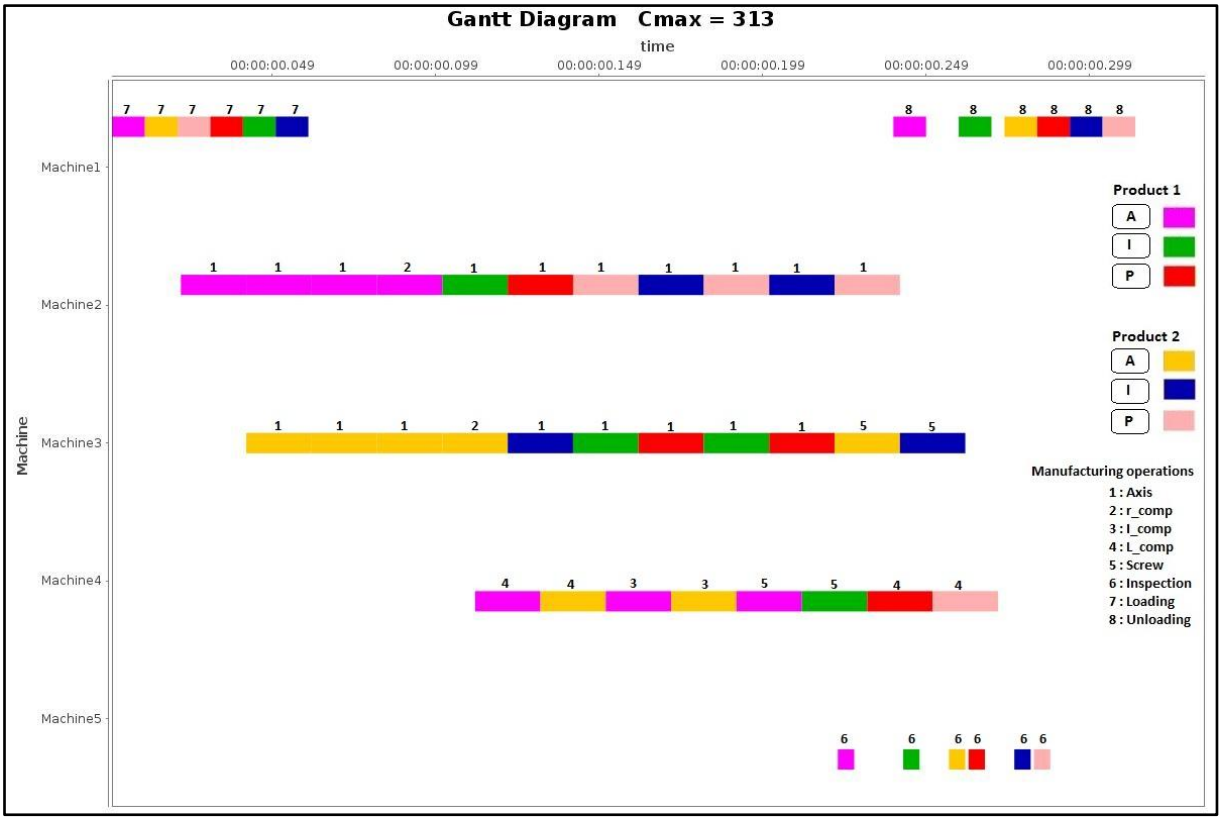


Figure 0.11 Diagramme de Gantt pour 2xA-I-P (IGA)

La figure 3.11 présente l'ordonnancement des opérations pour l'ordre de production 2 X A-I-P. Au niveau de la machine 1, il existe deux courtes fenêtres de temps perdu. Ceci est du au temps de transport entre les deux machine 5 et 1. Les deux machine 2, 3 sont parfaitement exploitées. La machine 4 attend l'arrivée du produit 1 « A » de la machine 2. Par contre, nous trouvons quelques fenêtres où la machine 5 est inexploitée, à cause du temps de transport entre les machines d'assemblage et la machine 5. La solution présentée sur la **Figure 0.11** est la meilleure solution parmi les autres méthodes présentées sur la **Table 0.7** et elle est très proche de la solution optimale.

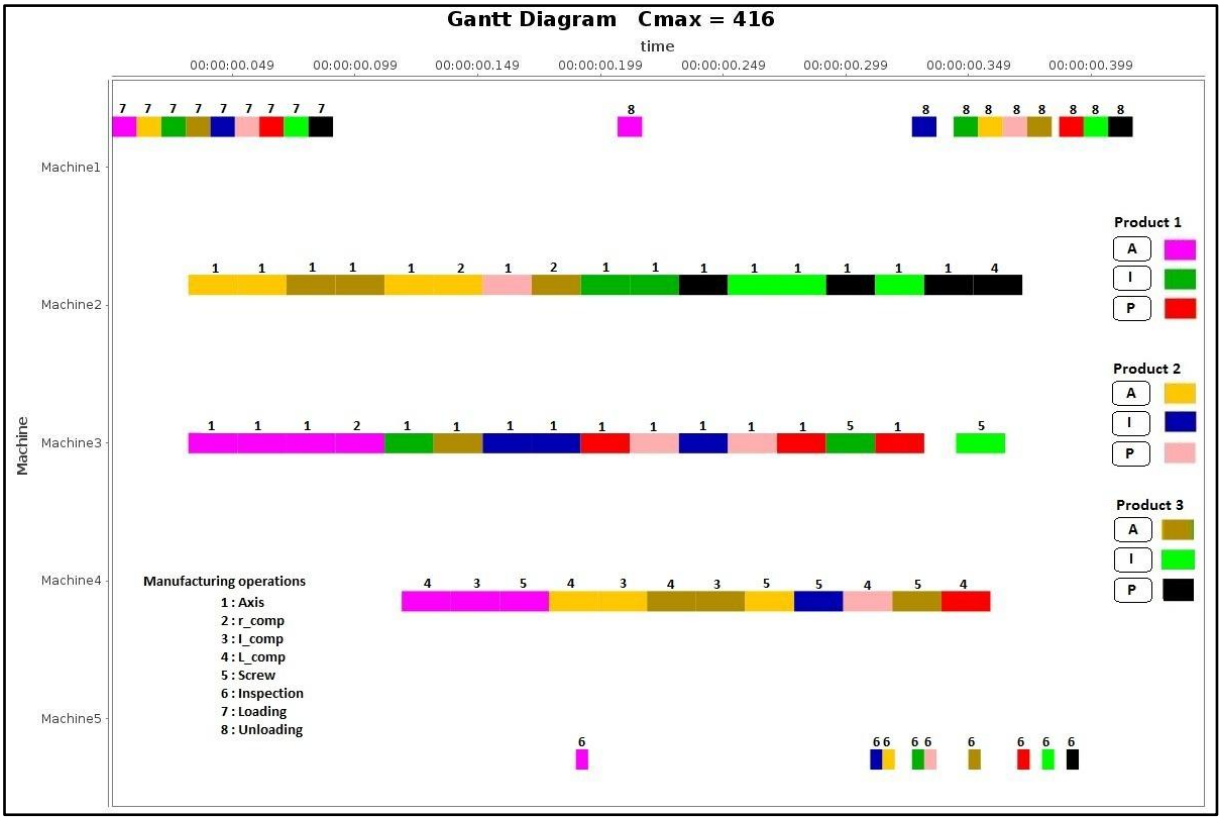


Figure 0.12 Diagramme de Gantt pour 3xA-I-P (IGA)

La solution présentée dans la **Figure 0.12** montre l'ordonnancement des opérations pour assurer la fabrication de trois produits 3 x A-I-P. La charge de travail est bien équilibrée entre les deux machines 2 et 3. Par contre ces deux machines sont plus chargées par rapport aux autres machines. Ceci est dû à cause de la nature d'opérations assurées par chaque machine. En fait, La majorité des opérations d'assemblage sont assurées par les deux machines 2 et 3.

Il existe une seule fenêtre où la machine 3 attend l'arrivé du produit 3 « I » de la machine 2. La majorité de ce type de fenêtres se trouvent au niveau de la machine d'inspection 5 en attendant l'arrivage des produits à inspecter. La performance de cette solution est presque identique à celle trouvée par la méthode ORCA-FMS (**Table 0.6**).

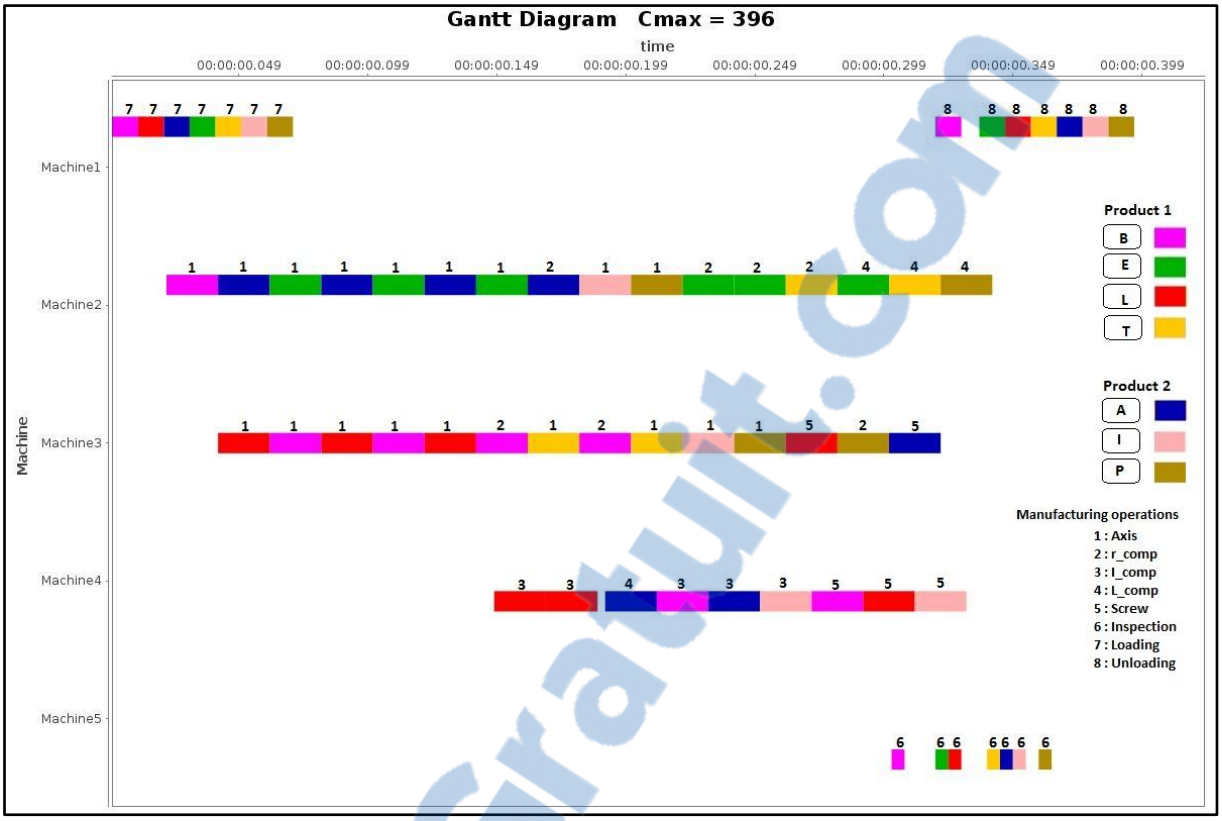


Figure 0.13 Diagramme de Gantt pour B-E-L-T-A-I-P (IGA)

La **Figure 0.13** présente l'ordonnancement des opérations pour la compagne de production B-E-L-T-A-I-P. Les deux machines 2 et 3 sont bien exploitées avec juste deux fenêtres où cette machine est inexploitée (temps d'arrivée des produits de la machine 1 de chargement des plateaux). La machine 4 attend l'arrivé du produit 1 « L » de la machine 3. Apres, il existe une courte fenêtre ou la machine 4 attend l'arrivée du produit 2 « A » de la machine 2. La machine 5 attend l'arrivée des produits à inspecter pour quatre reprises. Une seule fenêtre est présente au niveau de la machine 1en attendant l'arrivé du produit 1 « E » de la machine 2.

9. CONCLUSION

Nous avons utilisé l'algorithme génétique pour le problème d'ordonnancement du job shop flexible en prenant en considération le temps de transport des produits entre les machines. Le cas d'étude considéré reflète une cellule de production réelle. En revanche, les algorithmes génétiques ne peuvent pas assurer un optimum local. Pour faire face à cet obstacle, nous avons intégré la recherche Tabou, qui est bien adaptée pour trouver les solutions de voisinage. Ceci a permis d'améliorer la performance de l'algorithme génétique.

La fonction de voisinage proposée est plus efficace que les fonctions traditionnelles, basées sur des transitions aléatoires. Un grand nombre de ces transitions sont obsolètes et ralentissent la vitesse d'achèvement d'une solution de bonne qualité. La fonction de voisinage proposée est basée sur deux algorithmes concurrents, qui exploitent profondément les informations sur les opérations à optimiser.

Dans le chapitre suivant, nous prenons en considération la consommation de l'énergie pendant la production. Le problème initial devient un problème d'optimisation multi-objectif.

CHAPITRE 4 : UNE APPROCHE PREDICTIVE REACTIVE POUR LE CONTROLE D'ENERGIE POUR LE JOB SHOP FLEXIBLE

1. INTRODUCTION

Le développement technologique a permet d'améliorer le mode de vie des êtres humains et d'intensifier la production dans divers domaines. Mais, ce développement s'est répercuté négativement sur l'environnement. Donc, il est très important de prendre en considération l'aspect environnemental durant le développement des nouvelles technologies destinées à améliorer l'efficacité de la production. Dans ce contexte, la consommation de l'énergie est l'un des aspects les plus importants pour l'environnement. Par exemple, une production avec un pic d'énergie élevé peut augmenter les émissions du Co_2.

L'efficience de l'énergie des machines-outils[2] est généralement moins de 30 %. Par contre, Même si la consommation de l'énergie par les machines durant la fabrication représente une petite portion durant le cycle de vie de production, réduire l'énergie consommée durant la fabrication a été considérée récemment comme une des stratégies les plus importantes afin d'améliorer la durabilité dans la fabrication (Pušavec, Krajnik et Kopac 2010) (Prabhu, Trentesaux et Taisch 2015).

L'ordonnancement joue un rôle crucial dans l'efficacité de la production et a un effet sur la consommation des ressources ainsi que les émissions (He 2007). Donc, l'optimisation de l'ordonnancement des opérations sur les machines est très importante pour réduire la consommation de l'énergie, sans oublier le critère traditionnel du temps de production (Luo, et al. 2013).

L'efficience de l'énergie et l'efficacité des systèmes manufacturiers dans l'optimisation :

L'ordonnancement du problème job shop flexible est connu comme NP-difficile de point de vue complexité. Mais, Prendre en considération l'énergie dans l'ordonnancement des opérations manufacturières complique considérablement le problème classique. A cet effet, l'efficience doit être prise en considération en plus de l'efficacité. L'efficience vise à chercher la meilleure utilisation des moyens mais, l'efficacité a pour but de trouver les meilleurs résultats (Roghanian, Rasli et Gheysari 2012). En revanche, l'efficacité et l'efficience sont des objectifs contradictoires, à titre d'exemple réduire la consommation de l'énergie peut générer une perte au niveau de performance de temps de production (makespan).

Ce chapitre est organisé comme suit : dans un premier temps, nous présentons un état de l'art sur le concept de durabilité (Sustainability) dans l'ordonnancement des opérations manufacturier et l'importance de l'énergie dans ce contexte. La section suivante sera consacrée à la méthode prédictive proposée en utilisant NSGA-2. Ensuite, nous présentons la méthode réactive développée, destinée à contrôler le pic de l'énergie. La dernière section

[2] Une **machine-outil** est un équipement mécanique destiné à exécuter un usinage, ou autre tâche répétitive, avec une précision et une puissance adaptées. C'est un moyen de production destiné à maintenir un outil fixe, mobile, ou tournant, et à lui imprimer un mouvement afin d'usiner ou déformer une pièce ou un ensemble fixé sur une table fixe ou mobile. https://fr.wikipedia.org/wiki/Machine-outil

portera sur l'expérimentation en évaluant les résultats obtenus et en présentant une comparaison avec une autre étude

2. ETAT DE L'ART SUR L'ORDONNANCEMENT ET LE CONTROLE DE L'ENERGIE

Récemment, les études sur l'optimisation des systèmes manufacturiers en prenant en considération la contrainte de l'énergie, se sont multipliées. Ce type de problème est connu sous l'appellation Energy-aware manufacturing operations. Selon (Prabhu, Trentesaux et Taisch 2015), energy-aware des opérations manufacturières est un concept émergeant et très complexe. Ils ont défini ce concept comme « *un système de gestion des opérations manufacturières qui considère de manière prédictive ou réactive, outre les variables de décision de production habituelles, les objectifs et contraintes (temporels ou qualitatifs), l'énergie comme variable de décision, objective ou une contrainte* ».

Trois principaux challenges peuvent être trouvés en tenant compte l'énergie dans les systèmes manufacturiers (Prabhu, Trentesaux et Taisch 2015): 1. Energie-efficience vs efficacité des systèmes manufacturiers, 2. Volatilité de la disponibilité de l'énergie, l'offre et le coût, 3.Modélisation de la consommation d'énergie. Nous considérons dans cet article les deux premiers défis.

Dans le challenge 1 (énergie-efficience vs efficacité des systèmes manufacturiers), deux indicateurs de performance devraient être considérés. En revanche, prendre en considération l'énergie dans la gestion des systèmes manufacturiers, augmente significativement la complexité des problèmes classiques (e. g. le problème d'ordonnancement des ateliers est déjà classé comme NP-difficile). (Roghanian, Rasli et Gheysari 2012) ont définit l'énergie-efficience comme la meilleure utilisation des moyens or, l'efficacité signifie la recherche des meilleurs résultats. Par contre, l'efficience et l'efficacité sont généralement des objectifs contradictoires; Par exemple, la performance de l'ordonnancement des opérations manufacturières en terme de temps d'achèvement, peut être dégradée, lors de la réduction de l'énergie consommée. Il existe trois stratégies pour faire face à ce challenge (C. Pach, T. Berger, et al. 2014) : *1*. Energie-efficience peut être optimisée en considérant l'efficacité des opérations manufacturières comme une contrainte (e. g. économiser l'énergie d'une manière opportuniste : machine on/off). *2*. Optimiser l'efficacité des opérations manufacturières sous contrainte d'énergie (e. g. optimiser l'efficacité sous la contrainte de l'énergie totale disponible). *3*. Rechercher un compromis entre l'énergie-efficience et l'efficacité des opérations manufacturières.

La volatilité de l'énergie disponible, l'approvisionnement et le coût de l'énergie sont les principaux facteurs du challenge 2 dans Energy-aware des systèmes manufacturiers. Ces facteurs nécessitent des systèmes de gestion plus réactifs (Pouya, Sami et Bernard 2015). L'émergence de nouvelles sources d'énergie (e. g. panneau photovoltaïques, des éoliennes) a rendu le coût et la disponibilité de l'énergie plus imprévisibles. Par conséquent, le prix de

l'énergie devient plus dynamique. Tous ces facteurs, sont devenus des sources d'incertitude dans l'environnement de fabrication. (Christian, et al. 2002) ont classé l'incertitude en trois catégories en fonction du niveau de connaissance:
1. Incertitude totalement inconnue (e. g. accident sur le lieu de travail), 2. Doutes sur l'avenir (dans notre cas, nous tenons compte de la volatilité de l'approvisionnement énergétique) et 3. Incertitudes partiellement connues (e. g. panne des machine). Cela nous pousse à analyser les différentes approches de l'ordonnancement pour l'optimisation des systèmes manufacturiers, afin de faire face aux deux premiers challenges, cités précédemment.

Une classification intéressante de l'ordonnancement sous l'incertitude a été proposée par (Tarek, et al. 2014). Généralement, les chercheurs adoptent soit des approches prédictives ou réactives pour résoudre le problème d'energy-aware dans les systèmes manufacturiers. Les approches adoptées peuvent être qualifiées d'exactes ou approximatives.

Les premières approches largement proposées étaient exactes et prédictives (ordonnancement hors ligne), centrées sur l'optimisation de la consommation d'énergie, en utilisant des modèles typiquement mathématiques.

(K. Fang, N. A. Uhan, et al. 2013) ont optimisé la durée totale d'achèvement des opérations avec une contrainte de consommation d'énergie (énergie-efficience). Deux formulations basées sur la méthode exacte MIP (Mixed Integer Programming) ont été développées. Plusieurs autres méthodes exactes ont été proposées:, integer linear programming (Wang, et al. 2011) (Zhang, et al. 2012), mixed integer linear programming (Tonelli, et al. 2012). Le pic de l'énergie consommée était le premier aspect d'énergie abordé. L'approche développé par (Bruzzone, et al. 2012) était basée sur deux phases. Ils ont économisé l'énergie dans la seconde phase et l'ordonnancement initial était redéfinit par mixed integer linear programming, afin de contrôler le pic de l'énergie consommé. L'émission du Co2 a été prise en considération avec la minimisation de l'énergie consommée (K. Fang, N. Uhan, et al. 2011).

En revanche, du à la forte consommation du temps par ces modèles mathématiques, une seconde génération est apparue, basée sur des heuristiques ou méta-heuristiques (e. g. Recherche Tabou, algorithme génétique, etc.). Ces méta-heuristiques ou méthodes approximatives, permettent de trouver des solutions de bonne qualité dans un lapse de temps raisonnable. La combinaison des méthodes exactes avec des méthodes approximatives était proposée par plusieurs chercheurs. (Yildirim et Mouzon 2012) ont proposé une technique pour économiser la consommation de l'énergie. Ils ont arrêté les machines quand elles ne traitent pas des opérations. Un modèle mathématique a été proposé pour contrôler la consommation de l'énergie et la charge d'une seule machine. Un algorithme génétique multi-objectif était utilisé pour avoir un ensemble de solutions non dominées. En revanche, cette approche était limitée à une seule machine.

(Luo, et al. 2013) ont proposé un algorithme de colonie de fourmis pour le problème d'ordonnancement du flow-shop. Les fonctions objectif considérées sont la durée de

réalisation des opérations ainsi que le coût de l'électricité avec des différent prix relatifs aux périodes d'utilisation.

(Dai, et al. 2013) ont proposé une méthode pour un problème multi-objectifs pour l'éco-énergie avec le makepan et la consommation de l'énergie comme fonctions objectifs. Les auteurs ont proposé un modèle mathématique basé sur un mécanisme éco-énergétique (Enegy-efficient) pour le problème d'ordonnancement du flow shop. La technique de la somme pondérée a été utilisée afin d'obtenir des solutions de front de Pareto et un algorithme génétique-Recuit amélioré, est utilisé pour résoudre le problème multi-objectifs. Les tests appliqués sur un atelier de métallurgie ont montré la capacité de l'approche à identifier des solutions optimales du front de Pareto et que la relation entre le makespan et la consommation de l'énergie était conflictuelle.

Dans (Jiang et Zuo 2014), un modèle d'optimisation multi-objectif pour le problème du job shop flexible. Quatre fonctions objectif ont été prises en considération : la consommation de l'énergie, la durée de réalisation des opérations (makespan), le coût de traitement et le coût pondéré de qualité. NSGA-2 est utilisé. Afin d'éviter le problème de convergence prématurée, les auteurs ont proposé une technique appelée Blood Variation permettant d'améliorer les opérateurs génétiques de croisement et mutation.

Un modèle d'optimisation du coût de l'énergie a été développé pour une seule machine (Shrouf, et al. 2014). Les auteurs ont réduit les coûts de l'énergie tout en évitant les périodes dans lesquelles, le prix de l'utilisation de l'énergie est élevé. La prise des décisions est faite au niveau des machines : les temps de débuts de traitement des jobs, quand la machine doit être éteinte et démarrée. Le modèle proposé permet de préparer un ordonnancement avec le plan de production le moins cher.

Les approches méta-heuristiques proposées dans cette seconde génération sont plus rapides que les approches exactes. En réalité, un modèle mathématique ne convient pas à la réactivité demandée dans un contexte dynamique et incertain (Marik et McFarlane 2005). Cependant, les auteurs dans la seconde génération, ne prêtaient pas trop d'attention au contexte dynamique de l'Energy-aware des systèmes manufacturiers. Par exemple, ils se focalisent seulement sur un pic statique d'énergie consommée et ne gèrent pas le contrôle de la consommation d'énergie en temps réel. Comme suggéré par (Trentesaux et Prabhu 2014) pour un ordonnancement dans un contexte hautement dynamique, les approches réactives sont prometteuses pour la consommation et la limitation totale de l'énergie. (Prabhu et Taisch 2012) ont contrôlé la consommation de l'énergie par une heuristique nommée : "*distributed arrival time control*" pour une seule machine. (C. Pach, T. Berger, et al. 2014) ont proposé un modèle basé sur la technique des champs potentiels afin de réduire d'une manière réactive l'énergie totale d'un système manufacturier. Par contre, cette étude s'est limitée à la perte d'énergie sans faire attention au contrôle de la consommation globale d'énergie en fonction de l'approvisionnement électrique. Plus tard, (C. Pach, T. Berger, et al. 2015) ont proposé une méthode réactive basée sur des champs potentiels, en tant qu'extension du travail précédent pour l'ordonnancement des opérations dynamiques. Cette technique considère à la fois

l'efficacité et l'efficience. Les décisions d'assignement et de routage des opérations sont prises au niveau des produits, en fonction des champs potentiel attractifs émis par les machines. Les champs potentiels émis par les produits déterminent l'état des machines. Dans une autre étude visant les ateliers flow shop, les chercheurs (Huang, Yu et Chen 2017) proposent un algorithme génétique hybride pour minimiser le tarif d'énergie selon la période d'utilisation. Trois états de machines sont considérés, fonctionnant selon différente cadences.

D'après ces travaux, la majorité de ces méthodes sont soient prédictives or réactives. Les méthodes prédictives calculent un ordonnancement off-line, souvent quasi-optimal ou optimal. En revanche, un ordonnancement off-line peut rapidement devenir infaisable. Dans ces situations, les méthodes réactives sont utilisées dans les environnements distribués avec de fréquentes incertitudes. Mais, ces méthodes privilégient les décisions rapides au détriment de la qualité et elles ont de difficultés d'ordonnancement. Les approches prédictives-réactives sont considérées souvent comme approche supportant les risques comme des perturbations d'énergie. Dans ce contexte, nous proposons une approche prédictive-réactive pour le contrôle d'énergie dans les ateliers de production flexibles.

Notre approche est basée sur deux étapes :

1. une méthode prédictive : en utilisant NSGA-2 (Non Dominant Sorting Genetic Algorithm). Nous considerons trois fonctions objecctives pour évaluer les solutions :

- Le Cmax - La consommation totale de l'énergie - Le pic de l'énergie

2. Une méthode réactive : nous proposons un algorithme pour contrôle de pic de l'énergie d'une manière réactive, apres avoir lancé le plan de l'ordonanncement obtenu par NSGA-2. Les résultats obtenus ont été comparé à ceux obtenus par (C. Pach, T. Berger, et al. 2014).

Dans la section suivante, nous présentons la méthode préventive basée sur NSGA-2.

3. UNE METHODE PREDICTIVE–REACTIVE POUR LE CONTROLE DE L'ENERGIE

3.1. Une méthode prédictive pour le contrôle de l'énergie basée sur NSGA-2

Nous proposons une méthode préventive pour le contrôle de l'énergie en utilisant NSGA-2 (Non Dominant Sorting Genetic algorithm). La méthode est appliquée sur le job shop flexible. Trois fonctions sont choisies pour l'évaluation des individus : Cmax (makespan), la consommation totale de l'énergie et le pic de l'énergie. Donc, le problème peut être considéré comme un problème d'optimisation multi-objectifs. Une fois que l'ensemble de Pareto qui englobe plusieurs individus est trouvé, le décideur peut déterminer les poids de différentes fonctions objectifs par AHP (Analytic Hierarchy Process (Saaty 2008)) selon la situation actuelle de la production :

$$Obj_i = W_C \frac{C_{max} - C_i}{C_{max} - C_{min}} + W_E \frac{E_{max} - E_i}{E_{max} - E_{min}} + W_P \frac{P_{max} - P_i}{P_{max} - P_{min}} \quad \text{Equ. 0-1}$$

W_C, W_E, W_P sont les coefficients de cmax, consommation de l'énergie et le pic de l'énergie consommée.

C_i, E_i, P_i sont les valeurs de cmax, consommation de l'énergie et le pic de l'énergie de la solution i parmi une population de solutions.

3.1.1. Le modèle de Contrôle d'énergie

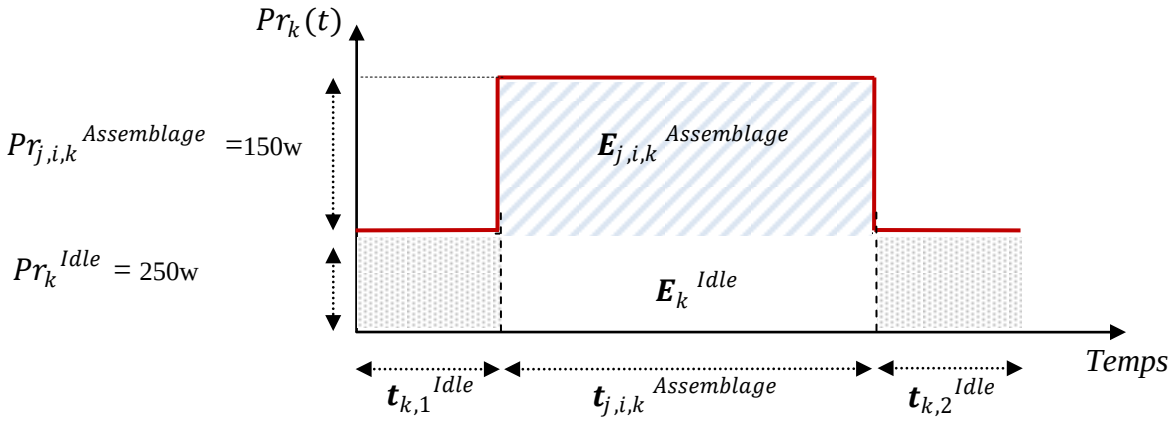


Figure 0.1 La puissance électrique d'entrée simplifiée d'une machine d'assemblage

Un modèle de consommation électrique d'une machine d'assemblage pour la cellule de production réelle AIP-PRIMECA est développé. Ce modèle est basé sur deux niveaux de consommation d'énergie: pendant le temps d'inactivité (mode veille) et lors d'une opération d'assemblage. L'évolution de la puissance électrique d'entrée requise dans le temps d'une machine M_k est déterminée par la fonction $Pr_k(t)$. Lorsqu'une machine M_k effectue une opération d'assemblage ou chargement/déchargement du plat d'assemblage, la puissance d'entrée requise est:

$$Pr_{j,i,k}^{Total} = Pr_{j,i,k}^{Idle} + Pr_{j,i,k}^{Assembling} \quad \text{Equ. 0-2}$$

- Pr_k^{Idle}: le niveau de puissance électrique [3] de la machine M_k en mode veille (Watts).
- $Pr_{j,i,k}^{Assembling}$: le niveau de puissance électrique de la machine M_k durant l'assemblage ou chargement/déchargement de l'opération $O_{j,i,k}$.
- $t_{j,i,k}^{Assembling}$: temps d'assemblage requis par la machineM_k. $t_{k,1}^{Idle}$, $t_{k,2}^{Idle}$ sont les périodes d'inactivité de la machineM_k.

[3] La puissance électrique Pe reçue par un appareil électrique est égale au produit de la tension électrique U entre ses bornes et de l'intensité I du courant électrique qui le traverse

- $E_{j,i,k}^{\ Assembling}$: l'énergie consommée par la machineM_k, pendant l'assemblage ou chargement/déchargement de l'opération $O_{j,i,k}$ (Watts-hour).
- $E_k^{\ Idle}$: l'énergie consommé par la machine M_k en mode veille ou standby (Watts-hour).

Selon les données de la cellule AIP-PRIMECA, la puissance électrique requise selon les trois modes (veille, assemblage, chargement/déchargement) est définie comme suit :

$Pr_{j,i,k}^{\ Idle}$ = 250 Watts , $Pr_{j,i,k}^{\ Assembling\ ,Chargement\ /déchargement}$ = 150 Watts

Supposant que $t_{k,s}^{\ Idle}$ est la période S d'inactivité de la machine Mk. L'énergie consommée par la machine M_k durant le temps d'inactivité S est définie comme suit:

$$E_{k,s}^{\ Idle} = Pr_k^{\ Idle} * t_{k,s}^{\ Idle} \qquad \text{Equ. 0-3}$$

L'énergie requise durant l'assemblage de l'opération $O_{j,i,k}$ est définie par :

$$E_{j,i,k}^{\ Assembling} = Pr_{j,i,k}^{\ Assembling} * t_{j,i,k}^{\ Assembling} \qquad \text{Equ. 0-4}$$

Par conséquent, l'énergie totale requise pour prendre en charge une opération (assemblage ou chargement/déchargement d'un plat d'une opération $O_{j,i,k}$) est définie par :

$$E_{j,i,k} = E_k^{\ Idle} + E_{j,i,k}^{\ Assembling} \qquad \text{Equ. 0-5}$$

L'énergie totale consommée par la machine M_k est définie par :

$$E_k = \sum E_{j,i,k}^{\ Assembling} + \sum E_{k,s}^{\ Idle} \qquad \text{Equ. 0-6}$$

Le modèle de l'ordonnancement est basé sur trois fonctions objectif : la durée de production (makespan ou Cmax), la consommation totale de l'énergie et le pic de l'énergie. Donc, l'optimisation de l'ordonnancement peut être représentée comme suit : Min (C, E, P). C représente le Cmax, E représente la consommation totale de l'énergie et P est le peak de l'énergie.

1. *Cmax* : la durée totale de traitement de tous les jobs.

2. *La consommation totale de l'énergie*

C'est l'énergie consommé par chaque machine durant la production. Donc, l'énergie totale est la somme des énergies consommé par les machines.

$$E^{Total} = \sum_{k=1}^{M} E_k \qquad \text{Equ. 0-7}$$

Ou, M est le nombre des machines présentes dans l'atelier.

3. *Le Pic de l'énergie* :

C'est la consommation maximale de l'énergie.

$$P_{peak} = \max_{t=1,..Cmax} E_t \qquad \text{Equ. 0-8}$$

E_t est la consommation de l'énergie de toutes les machines à l'instant t :

$$E_t = \sum_{k=1}^{M} E'_{k,t} \qquad \text{Equ. 0-9}$$

Sachant que $E'_{k,t}$ est la consommation d'énergie par la machine M_k à l'instant t.

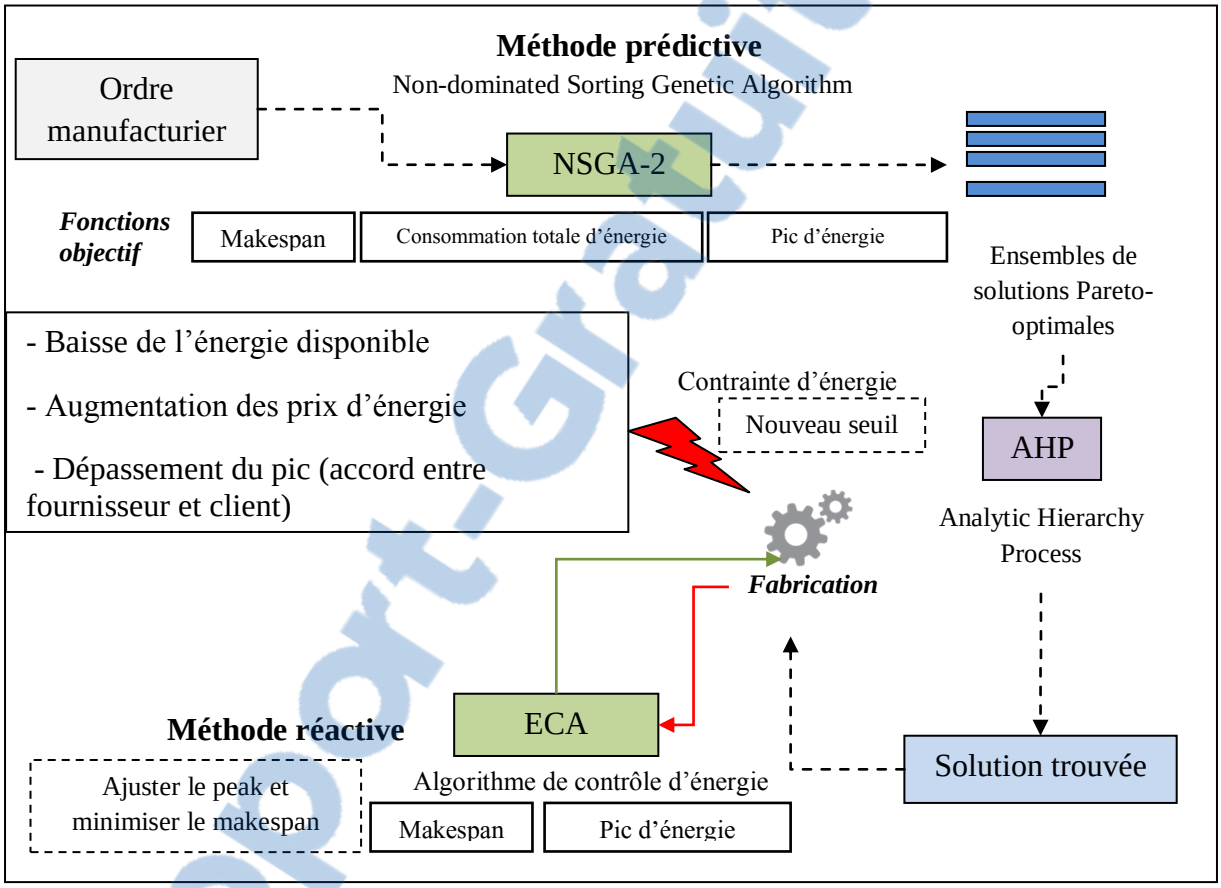


Figure 0.2 Le modèle prédictive-réactive proposé de le contrôle d'énergie

3.1.2. L'algorithme de l'ordonnancement : NSGA-2

Plusieurs méthodes pour l'optimisation multi-objectifs ont été proposées. Les méthodes utilisées sont : agrégation en une seule fonction objectif, pondération aléatoire, l'optimisation basée sur le front de Pareto (Xing, Chen et Yang 2009). La méthode d'optimisation de Pareto est préférée par les chercheurs car, elle permet d'obtenir un ensemble de solutions optimales de Pareto dans le processus de l'optimisation, qui est cohérent avec le problème d'ordonnancement considéré (Moslehi et Mahnam 2011). Plusieurs algorithmes évolutifs

peuvent être trouvés dans la littérature comme, MOGA (Multi Objective Genetic Algorithm), NSGA et NSGA-2 (Non Dominant Sorting Genetic Algorithm) (Srinivas et Deb 1995) (Deb, et al. 2002), PESA et PESA-2 (Pareto Envelope-based Selection Algorithm) (Corne 2000). En revanche, vu sa meilleure distribution, NSGA-2 est l'algorithme multi-objectif le plus utilisé.

a. Codage de chromosomes

Comme le montre la table 4.1, le codage d'une solution est basé sur les paires : machine-opération. Ce codage représente l'assignement et le séquencement des opérations.

MACHINE	2	1	3	3	...
OPERATION	$O_{3,1}$	$O_{1,1}$	$O_{3,2}$	$O_{3,3}$	...

Table 0.1 Codage des chromosomes

D'après la **Table 0.1**, la première opération qui sera lancée par la machine 2 est O3,1 (première opération du job 3), et O1,1 (première opération du job 1) au niveau de la machine 1 et ainsi de suite.

b. L'opération de sélection (Elitism Non Dominated Sorting Algorithm)

La nature de cette opération est plus complexe par rapport à celle d'un algorithme génétique mono objectif. Cette sélection (multi-objectifs) consiste généralement à classer les individus dans des rangs et appliquer une stratégie des solutions non dominées du même rang.

La procédure utilisée pour le classement des individus est décrite comme suit (Deb, et al. 2002):

La première étape consiste à calculer deux paramètres pour chaque solution P:

- Np : le nombre de solutions qui dominent la solution P (le nombre de domination),

- Sp : l'ensemble de solutions dominées par la solution P.

Au départ, les deux paramètres Np et Sp de toutes les solutions sont nuls. Pour chaque solution P, chaque membre q appartenant à Sp est visité une seule fois et son nombre de domination (Nq) est réduit par un (Nq-1).

En répétant cette étape, si le nombre de domination d'un membre q devient null (Nq = 0), ce membre sera mis dans une liste Q. Les Membres de cette liste (Q) appartiennent au deuxième front des solutions non dominées. Cette procédure est répétée sur chaque membre de la liste Q et le troisième front est identifié. Une fois que tous les fronts sont identifiés, la procédure est arrêtée.

Fast-Non-Dominated-Sort (P) (Deb, et al. 2002)

Begin
For each $p \in P$
  $N_p = 0$
  $S_p = \emptyset$
  For each $q \in P$
    If $(p < q)$ // if p dominates q
      $S_p = S_p \cup \{q\}$ // ajouter q à l'ensemble des solutions dominées par p
    Else Si $(q < p)$ // if q domine p
      $N_p = N_p + 1$ // incrémenter le nombre de domination de p
  If $(N_p == 0)$ // si p appartient au premier front
    $P_{rank} = 1$
    $F_1 = F_1 \cup \{p\}$
  End-Each
i = 1
End-Each // initialiser le compteur de front
While $(F_i \neq \emptyset)$
  Q = $\emptyset$ // une liste pour stocker les membres du prochain front
  For each $p \in F_i$
    For each $q \in S_p$
      $N_q = N_q - 1$
      If $(N_q = 0)$ // q appartient au prochain front
        $q_{rank} = i + 1$
        $Q = Q \cup \{q\}$
    End-each
  End-each
  i = i + 1
  F_i = Q
End-while
End

Apres avoir classé les solutions d'une population dans différents rangs, il faut classer les solutions du même rang. La technique appliquée est « *Crowding distance* ». L'utilité de cette technique est d'assurer la diversité de la population. Donc, les solutions ayants une valeur de Crowding distance importante seront favorisées par rapport aux autres solutions.

Le calcul de *Crowding distance* de chaque solution nécessite un tri croissant de la population, selon la valeur de chaque fonction objectif. Les solutions ayants la valeur minimale / maximale de chaque fonction objective seront toujours prises. Donc, ces solutions auront toujours une valeur infinie de Crowding distance. Les autres solutions intermédiaires auront une valeur de distance absolue et normalisée, égale à la différence des deux fonctions objectif des deux solutions adjacentes. Ce calcul de distance normalisée sera répété pour toutes les fonctions objectif.

La somme de ces distances normalisées de chaque fonction objectif d'une solution, nous donnera la valeur de Crowding distance de cette solution.

L'algorithme suivant (crowding-distance (I)) montre l'assignement de la valeur de Crowding distance à chaque solution.

Algorithme : Crowding-Distance (*I*) (Deb, et al. 2002)

Begin
L = $|I|$ // le nombre de solutions dans la liste *I*
For each i, set $I[i]_{distance} = 0$ // initialiser la distance de chaque solution
For each objective m
$I = sort\ (I, m)$ // trier selon la valeur de chaque fonction objectif
$I[1]_{distance} = I[L]_{distance} = \infty$ // les solutions frontalières sont toujours prises
For i = 2 **to** (L-1) // pour toutes les autres solutions
$I[i]_{distance} = I[i]_{distance} + \big((I[i+1] * m) - (I[i-1] * m)\big)/\big({f_m}^{max} - {f_m}^{min}\big)$
End-for
End-for
End-for
End

$I[i] * m$ est la valeur de la fonction objectif *m* de l'individu *i* dans l'ensemble de solutions *I*. ${f_m}^{max}$, ${f_m}^{min}$: la valeur maximale et minimale de la fonction objectif *m*.

La solution *i* est meilleure que la solution *j* si :

1. $i_{rank} < j_{rank}$ ou

2. $i_{rank} == j_{rank}$ et $i_{distance} > j_{distance}$

Si deux solutions appartiennent au même rang, la solution qui se trouve dans une zone moins comblée sera favorisée.

Algorithme principal de sélection

Apres la génération de la population initiale P_0, cette population est classée selon la méthode de non domination. Un rang est assigné à chaque solution, selon le niveau de non dominance (rang 1 est le meilleur rang, rang 2 est moins bon ainsi de suite). Au début, les individus qui vont participer aux opérateurs de recombinaison et mutation sont sélectionnés aléatoirement. Une population d'enfants Q_0 sera générée de taille N.

Les deux listes de la population P_t et des enfants Q_t de la génération *t* sont rassemblés ($R_t = P_t \cup Q_t$) (Table 4.2). La taille de la population R_t est 2N. La méthode des solutions non dominées sera appliquée sur R_t pour trier les solutions. De cette manière, le concept de l'élitisme est assuré.

Les solutions appartenant au premier Front F_1 sont les meilleures solutions parmi les solutions de la population P_t et celle des enfants Q_t. Si la taille de F_1 est inférieure à *N*, les solutions du prochain front F_2 seront sélectionnées et ainsi de suite, jusqu'à ce que *N* individus soient sélectionnés. Donc, la sélection des solutions est basée sur le rang et la valeur de *Crowding distance* de chaque solution.

L'algorithme principal de sélection est présenté comme suit :

Algorithme : sélection (Deb, et al. 2002)

Begin

$R_t = P_t \cup Q_t$ // combiner les parents et les enfants

Fast-Non-Dominated-Sort (R_t) // déterminer tous les fronts des solutions non dominées $F_1, F_2, \ldots$

$P_{t+1} = \emptyset$; i = 1

until ($|P_{t+1}| + |F_i| \leq N$) // jusqu'a ce que la population des parents soit complétée.

Crowding-Distance (F_i) // calculer Crowding distance dans F_i

$P_{t+1} = P_{t+1} \cup F_i$ // ajouter le i ème front dans la population des parents

i = i + 1 // tester le front suivant pour l'inclusion

Sort(F_i, Crowding-distance) // trier F_i selon un ordre décroissant sur la base de crowding distance

$P_{t+1} = P_{t+1} \cup F_i[1 : N - |P_{t+1}|]$ // choisir les premiers éléments ($N - |P_{t+1}|$) de F_i.

$Q_{t+1} = create\_new\_pop(P_{t+1})$ // appliquer sélection, croisement et mutation pour la nouvelle population

t = t + 1 // incrémenter le compteur de génération

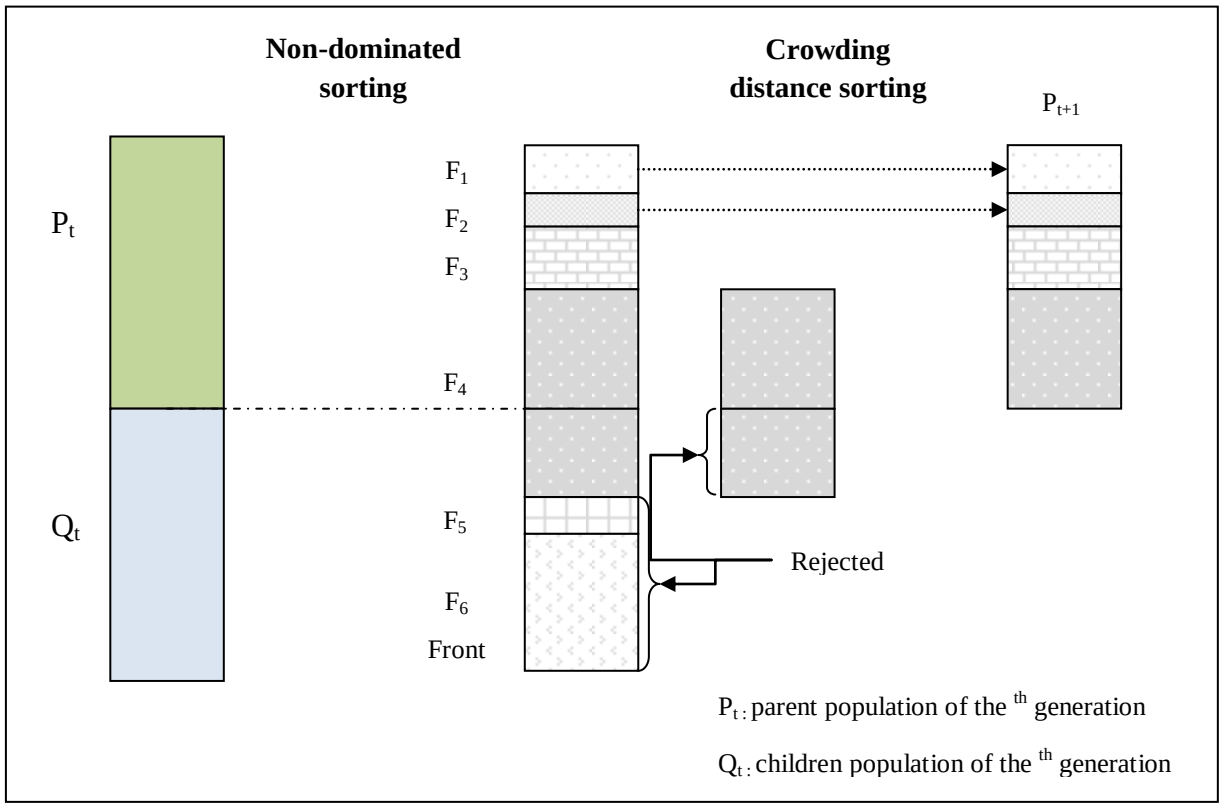


Figure 0.3 la procédure de NSGA-2 (Deb, et al. 2002)

c. Les opérateurs de croisement et de mutation

Ces deux opérateurs génétiques utilisés pour l'optimisation multi-objectifs, ont été implémentés de la même façon que l'algorithme génétique proposé dans le chapitre 3 et utilisé pour une optimisation mono objectif. Nous rappellons que l'opérateur de croisement

utilisé est POX (**Figure 0.2**), qui génère des solutions valides, ce qui nous permet d'éviter une procédure de correction des gènes.

Deux types de mutation sont implémentés : mutation pour le séquencement (routage) (**Figure 0.3**) et mutation d'assignement (**Figure 0.4**). Cet opérateur a pour but d'introduire une légère perturbation au niveau des gènes des solutions enfants afin de garantir la diversité et d'éviter une convergence prématurée de l'algorithme génétique.

3.2. Méthode réactive pour le contrôle de l'énergie

De nombreux facteurs sont apparus ces dernières années qui poussaient à la nécessité de réactivité des systèmes de gestion manufacturiers. Parmi ces facteurs, la volatilité et l'imprévisibilité de disponibilité de l'énergie ainsi que l'instabilité de l'approvisionnement et du coût de l'énergie (Ghadimi, Kara et Kornfeld 2015). A titre d'exemple, les émissions du carbone sont importantes durant les périodes des pics d'électricité à cause de l'utilisation des ressources plus chères et moins propres (V. Prabhu 2012).

Dans le même contexte, des accords d'approvisionnement de l'énergie entre le client et le fournisseur doivent être respectés et toute consommation supplémentaire pénalise le client. En plus, la tarification de l'électricité peut être dynamique.

Parmi d'autres facteurs apparus récemment, qui accentuaient l'imprévisibilité du coût de consommation de l'énergie ainsi que sa disponibilité et la volatilité de son approvisionnent est l'augmentation excessive de l'utilisation des panneaux solaires et les éoliennes comme des nouvelles sources d'énergie. L'évolution de disponibilité de l'énergie doit être prédite mais le prix, la charge et le comportement de consommation peut être difficile à prédire (Fan et Borlase 2009). A cet effet, la définition des stratégies exactes pour les systèmes manufacturiers dans ce contexte difficile à prédire, est une tâche difficile (Kuhlmann et Bauernhansl 2015).

Il existe plusieurs méthodes pour résoudre ce challenge telle que des outils d'analyse de données, des outils de gestion de risque, des outils de simulation ou des algorithmes réactifs (des heuristiques rapides, architecture multi-agents, etc.) (Prabhu, Trentesaux et Taisch 2015).

Dans ce contexte, nous proposons un algorithme réactif qui vise à contrôler le pic de la consommation de l'énergie ainsi que la durée de production (makespan). L'algorithme réactif vise à trouver un compromis entre la durée de production le pic de l'énergie consommée. Cet algorithme s'applique sur la solution générée par l'algorithme génétique NSGA-2 (méthode prédictive) afin de contrôler le pic de consommation de l'énergie.

Dans la section suivante, l'algorithme réactif de control d'énergie est présenté.

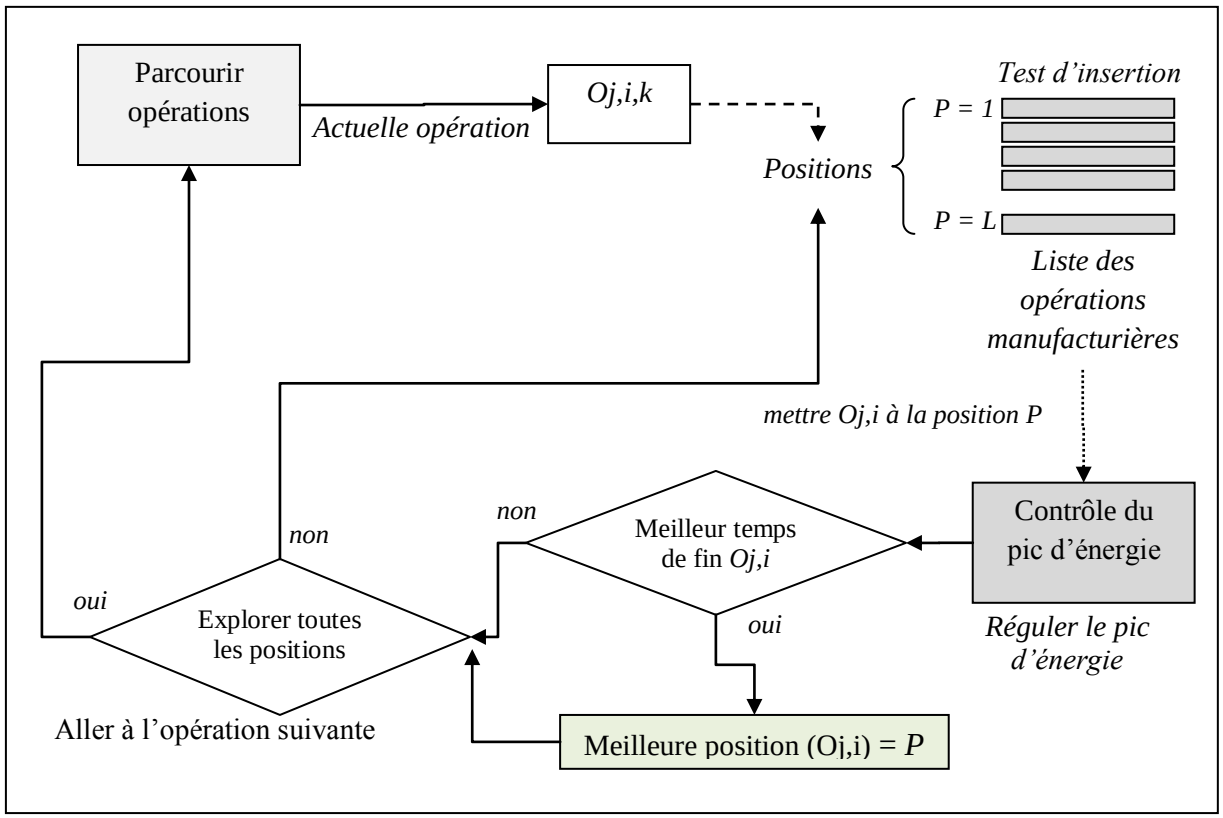


Figure 0.4 le modèle proposé de contrôle du pic l'énergie consommée

La **Figure 0.4** présente le modèle proposé de contrôle du pic de l'énergie consommée par les machines de production. Ce modèle est basé sur l'optimisation de l'ordonnancement des opérations afin de réguler le pic de l'énergie consommée, sans oublier l'optimisation de la durée de production (cmax).

Ce modèle de régulation de l'ordonnancement des opérations manufacturières peut intervenir dans plusieurs cas :

- une baisse de réapprovisionnement de l'énergie. Ce cas est très répondu à cause de l'utilisation de nouvelles sources de génération de l'énergie telle que : panneaux photo voltaïques, les éoliennes, etc. ces nouvelles sources de l'énergie sont écologiques mais l'approvisionnement est peu stable.

- Augmentation de prix de l'énergie dans des périodes : dans certains cas, le serveur de l'énergie peut exiger une augmentation de tarification, surtout dans les périodes ou le pic de consommation est très élevé.

- dépassement du pic d'énergie : dans le cas d'un contrat du pic d'énergie entre le serveur et le client, ce dernier se trouve dans l'obligation de respecter ce pic sinon, une pénalité peut être exigée, ce qui peut avoir une répercussion sur la rentabilité de l'entreprise.

Tous ces événements peuvent surgir à n'importe quel moment, ce qui nécessite une méthode réactive et rapide, afin de réguler l'ordonnancement des opérations pour faire face à une telle exigence énergétique.

A chaque fois qu'un pic d'énergie est exigé, le modèle réordonnance les opérations en se basant sur les étapes suivantes :

1. Insérer chaque opération dans une position et faire appel à une procédure qui régule le pic de l'énergie

2. choisir la meilleure position qui correspond au temps d'achèvement au plus tôt de l'opération en cours

3. Répéter les deux premières étapes jusqu'à ce que toutes les opérations soient parcourues.

A la fin, nous aurons un ordonnancement dont le pic est régulé et la durée de production est minimale.

L'algorithme *control_pic energy(S, peak)* recalcule l'ordonnancement des opérations pour limiter le pic de consommation de l'énergie selon le pic exigé. Cette procédure à deux entrées : l'ordonnancement initial et le pic exigé.

Cette procédure commence par tester le routage de chaque opération en l'insérant à une position. Apres, la boucle suivante recalcule le temps de début et de fin des opérations de telle sorte à réguler le pic de consommation de l'énergie. Nous répétons la procédure pour toutes les positions et celle qui permet d'avoir le temps d'achèvement le plus tôt sera retenue.

Algorithme : control_pic_energy(S, peak)

Begin
  For each $O_{j,i}$ // explorer chaque opération
    For P = 1 to L // explorer les positions de chaque opération, L est la dernière position
      Insert ($O_{j,i}$; P) // insérer $O_{j,i}$ dans la position P
      $tO_{j,i,k} = Max\left((tfO_{j,i-1} + transport\_time), tfO^{-1}{}_{j,i}\right)$ // temps du début
      $tfO_{j,i,k} = tO_{j,i,k} + P_{j,i,k}$ // temps de fin
      Shift = 0 // initialiser le compteur du décalage

      while $Min(t_2, tfO_{j,i,k}) - Max(tO_{j,i,k}, t_1) > 0$ && $peak\_interval(S, t_1, tfO_{j,i,k}) > peak$
        $tO_{j,i,k} = Max\left((tfO_{j,i-1} + transport\_time), tfO^{-1}{}_{j,i}\right)$ + Shift // nouveau temps du début
        $tfO_{j,i,k} = tO_{j,i,k} + P_{j,i,k}$ // le temps du début de $O_{j,i}$ est décalé par une seconde
        Shift = shift + 1; // incrémenter le compteur du décalage par une seconde.
End.

L'algorithme *peak_interval (S, t1, t2)* mesure le pic de consommation de l'énergie dans un interval temporel délimité par les deux bornes t1 et t2. Cet algorithme fonctionne avec une fonction nommée energy_consumption_peer_second qui mesure le pic de l'énergie à un instant donné.

```
Algorithme: peak_interval ( S, t1, t2 )
Begin
  current_peak = max_peak = 0
  for (i = t1 to t2 ) // explorer l'intervalle temporel entre t1 et t2
       current_peak = energy_consumption_peer_second (S , i , i+1)
       if (current_peak > max_peak )
          max_peak = current_peak

End.
```

L'algorithme *energy_consumption_peer_second* retourne le pic de l'énergie à un instant donné. Les machines en mode fonction consomment 400 watts et 250 watts à l'arrêt.

```
Algorithme: energy_consumption_peer_second (S , i , i+1)
Begin
  Energy_consumption = 0
  For each machine k
     For each operation Oj,i,k
        If ( current_machine = k)
           If(tO_{j,i} ≤ t_1  && tfO_{j,i} ≥ t_2)
              Energy_consumption = Energy_consumption + 400 // 400 est la consommation d'énergie
en mode assemblage
        If (Energy_consumption = 0)
          Energy_consumption = Energy_consumption +250 // est la consommation d'énergie en mode
veille
End.
```

3.2. Contrôle de perte d'énergie

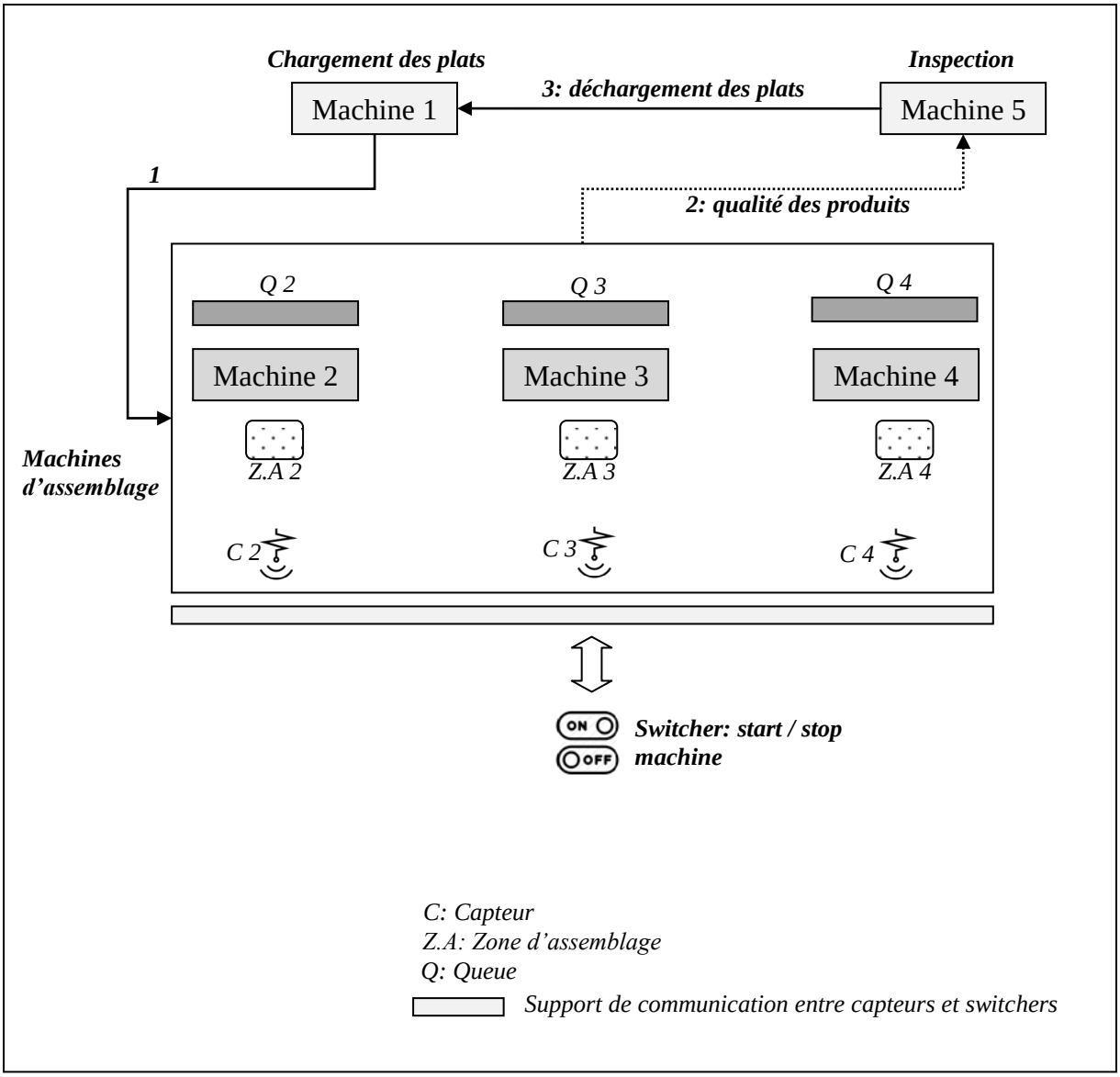


Figure 0.5 Modèle proposé pour contrôler la perte d'énergie

Après avoir présenté un modèle d'ordonnancement des opérations visant à contrôler le pic de consommation de l'énergie, nous proposons un autre modèle afin de limiter la perte de l'énergie consommée par les machines lorsqu'elles sont en attente des opérations.

Ce modèle est basé sur le cas d'étude présenté dans le chapitre 3 (AIP-PRIMICA cell). Cette cellule de production est composée de cinq machines :

- machine 1 : chargement des plats vides et déchargement des plats remplis par des produits finis.

- machine 2, machine 3 et machine 4 : c'est des machines d'assemblage des pièces afin d'obtenir des produits finis.

- machine 5 : lorsque les machines 2, 3 et 4 terminent l'assemblage, la machine 5 effectue un test de qualité des produits finis, si un défaut est détecté par cette machine, une opération manuelle de maintenance sera lancée. La maintenance est la seule opération manuelle dans la cellule de production. La maintenance n'est pas considérée dans cette étude.

La majorité des opérations sont effectuées par les machines d'assemblage. Ces dernières ont trois éléments :

- une liste d'attente (Queue) : contenant les produits non finis en attente pour l'assemblage.

- une zone d'assemblage (*zone d'assemblage*) : chaque produit est transporté par des convoyeurs automatiques, lorsque la liste d'attente d'une machine d'assemblage est vide, le convoyeur de produit transporté se positionne dans la zone d'assemblage.

- Capteur : le rôle du capteur est de tester si la liste d'attente ainsi que la zone d'assemblage sont libres ou pas. Un support de communication existe entre le capteur de chaque machine d'assemblage et le switcher.

Le switcher reçoit les informations émises par le capteur de chaque machine d'assemblage et réagit. Si la liste d'attente d'une machine ainsi que sa zone d'assemblage sont libres, le switcher arrête cette machine d'assemblage. Une fois qu'un produit est présent dans sa liste d'attente, le switcher démarre la machine. De cette manière, nous pouvons économiser l'énergie.

Donc, le contrôle de l'énergie peut se faire par l'ordonnancement des opérations et par le contrôle des machines. La combinaison des deux modèles est possible, mais pour des raisons de simplification, nous avons séparé les deux approches.

Insertion des opérations dans les fenêtres libres

Afin de limiter la dégradation de la qualité de l'ordonnancement généré par l'algorithme prédictive (NSGA-2) pendant le contrôle de l'énergie, nous proposons une heuristique (intégrée dans l'algorithme de contrôle du pic) visant à mieux exploiter les fenêtres temporelles. L'exemple suivant illustre le principe de cette heuristique.

Dans la **Figure 4.6**, nous constatons deux périodes inexploitées au niveau de la machine *M3* et même chose pour la machine *M1*.

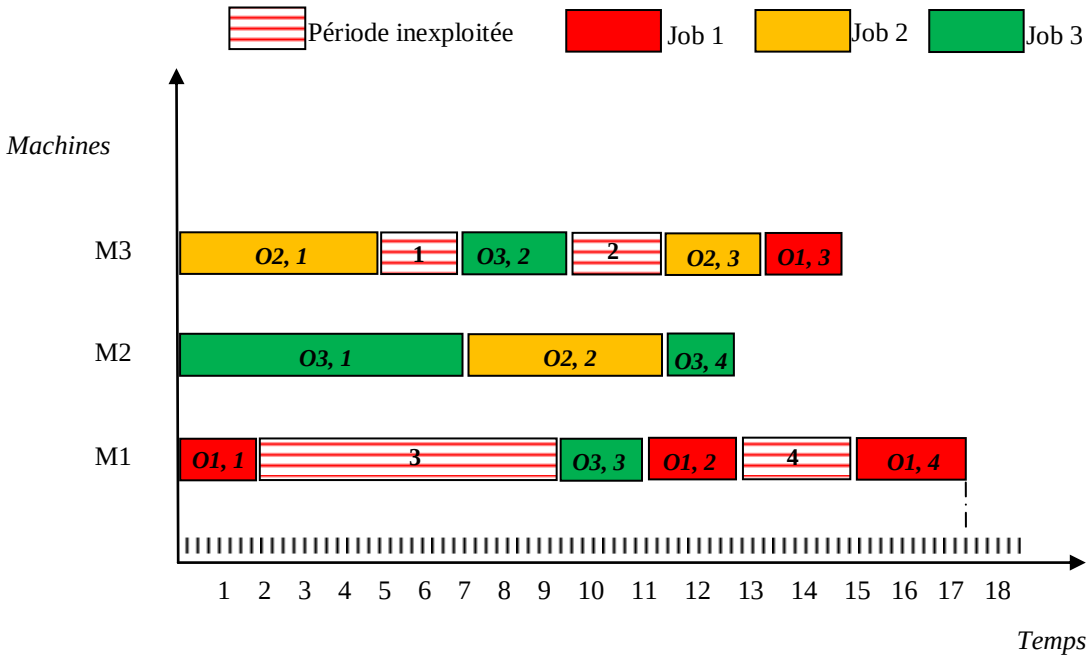


Figure 0.6 Ordonnancement avant l'optimisation

L'opération *O1, 4* est la dernière opération. Donc, nous favorisons les déplacements des opérations du *Job 1*.

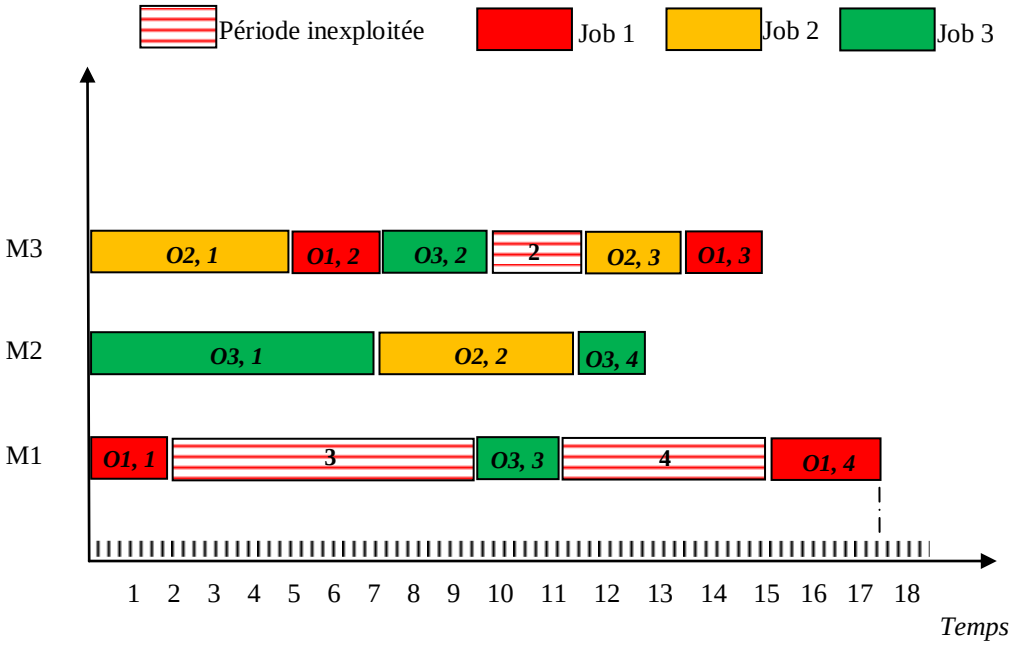


Figure 0.7 Ré assignement de l'opération O1, 2 (M1 vers M3)

L'opération *O1, 4* est la dernière opération. Donc, nous favorisons les déplacements des opérations du *Job 1*.

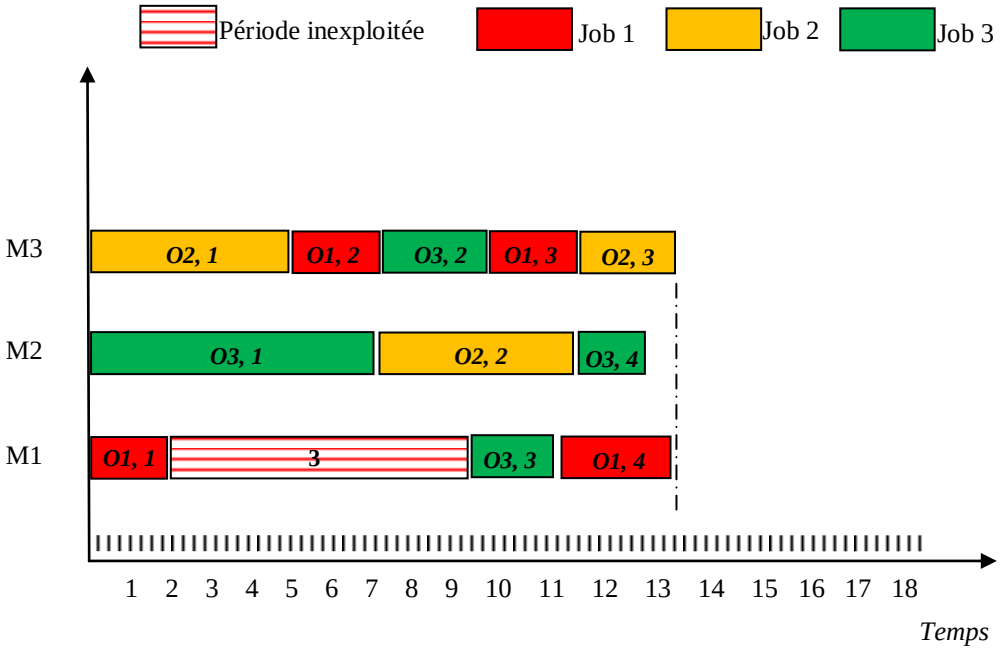


Figure 0.8 Ré séquencement de l'opération O1, 3

L'opération *O2, 3* est la dernière opération. Donc, nous favorisons les déplacements des opérations du *Job 2*.

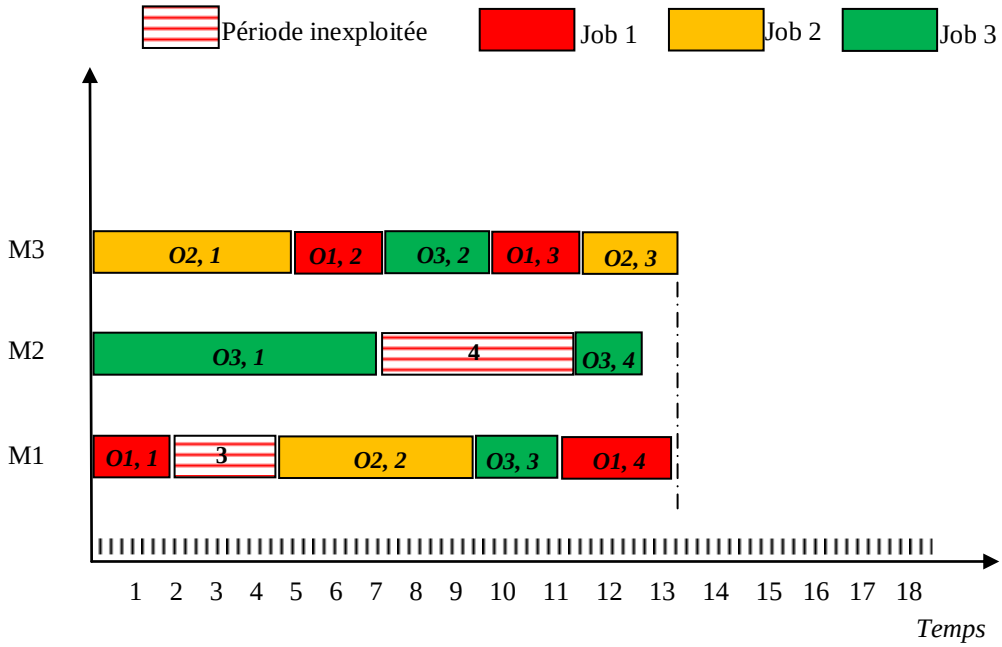


Figure 0.9 Ré assignement de l'opération O2, 2 (M2 vers M1)

L'opération *O2, 3* est la dernière opération. Donc, nous favorisons les déplacements des opérations du *Job 2*.

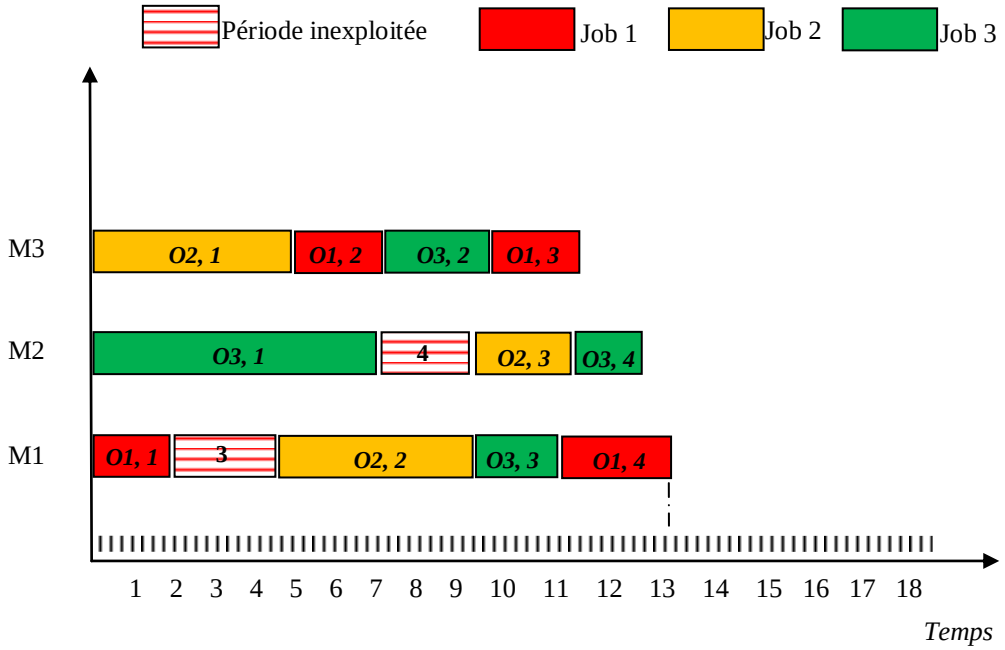


Figure 0.10 Ré assignement de l'opération O2, 3 (M3 vers M2)

L'opération *O1, 4* est la dernière opération. Donc, nous favorisons les déplacements des opérations du *Job 1*.

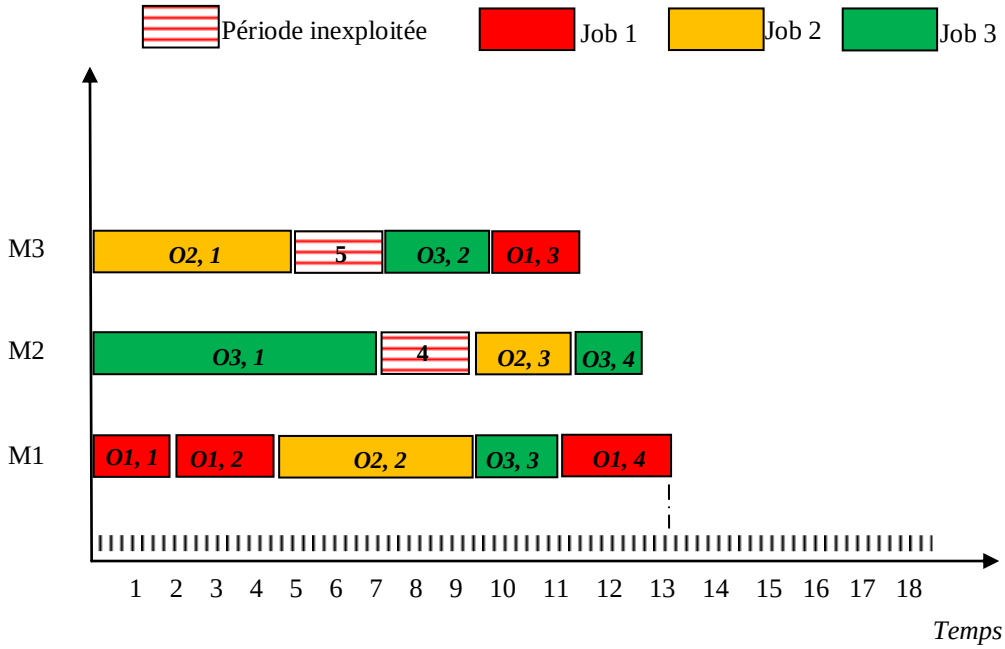


Figure 0.11 Ré assignement de l'opération O1, 2 (M3 vers M1)

L'opération *O1, 4* est la dernière opération. Donc, nous favorisons les déplacements des opérations du *Job 1*.

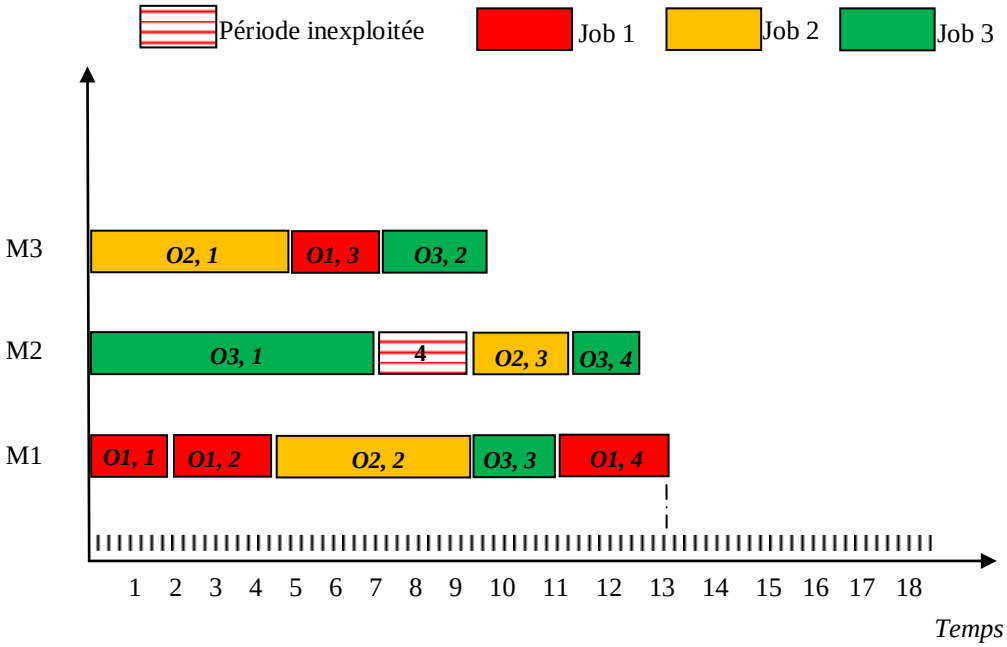


Figure 0.12 Ré séquencement de l'opération O1, 3

L'opération *O1, 4* est la dernière opération. Donc, nous favorisons les déplacements des opérations du *Job 1*.

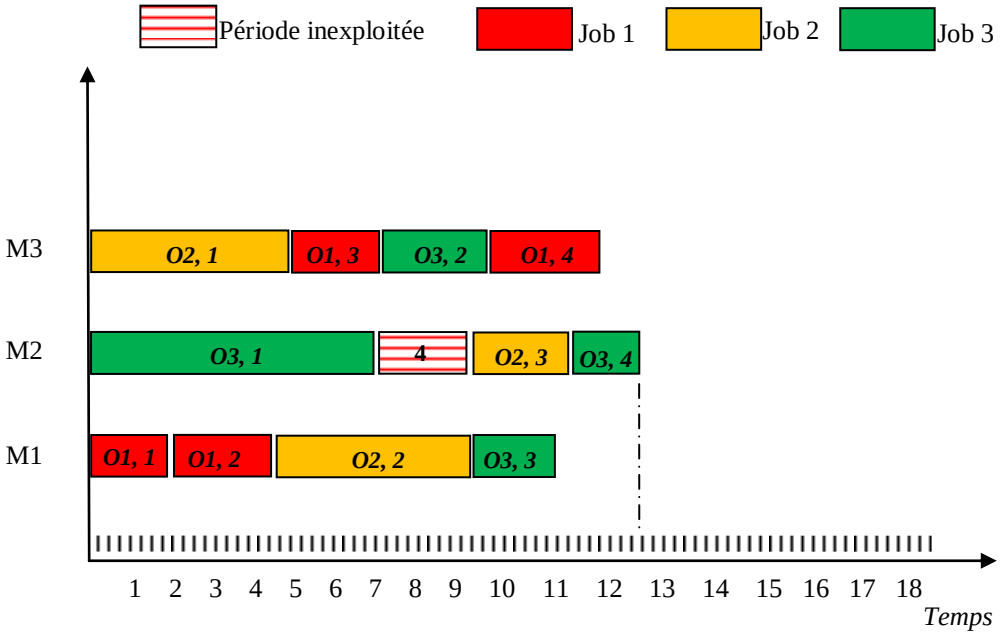


Figure 0.13 Ré assignement de l'opération O1, 4 (M1 vers M3)

L'opération *O3, 4* est la dernière opération. Donc, nous favorisons les déplacements des opérations du *Job 3*. Déplacement impossible pour le Job 3. Arrêt d'optimisation.

Rappelons que :

- l'opération *Oj,i* doit être avancée dans le temps : sa date de début doit être postérieure à la date de début de la période inexploitée,

- déplacement de *Oj,i*, ne doit pas entrer en conflit avec le prédécesseur et le successeur de l'opération par rapport à la gamme opératoire du Job. Dans la partie suivante, nous présentons le cas d'étude.

4. LE CAS D'ETUDE

Le modèle proposé pour le contrôle de l'énergie est appliqué sur la cellule de production (AIP-PRIMECA) de l'université de valenciennes (France), utilisée pour l'optimisation mono objectif (durée de production) dans le chapitre 3. Une modélisation formelle de cette cellule de production sous forme de benchmark a été présentée par (Trentesaux, et al. 2013) et toutes les données sont accessibles à travers un site internet[4]. Les données sur AIP-PRIMECA, décrites dans le chapitre 3 sont organisées sous trois parties :

- les données relatives aux produits

- les données relatives aux ressources

- les données du système de transport.

Vu que l'énergie est considérée dans ce chapitre, les données concernant la consommation de l'énergie des machines sont nécessaires. La **Table 0.2** présente la consommation électrique des machines (en appliquant le modèle de contrôle de perte d'énergie **Figure 0.5**). Les composants sont assemblés par les machine 2,3 et 4 l'assemblage est partiellement flexible. Donc, il y a une possibilité d'améliorer le contrôle de ces ressources vis-à-vis la consommation de l'énergie. Par contre, les deux machine 1 (chargement / déchargement) et machine 5 (inspection) sont obligatoire pour chaque produit, ce qui explique pourquoi nous avons considéré que les machines 2, 3 et 4.

Machines	***Consommation électrique (Watts)***		
	En veille	***en veille*** **Figure 0.5**	***fonctionnement***
Machine 2	250	0	400
Machine 3	250	0	400
Machine 4	250	0	400

Table 0.2 Consommation électrique des machines en watt

[4] http://www.univ-valenciennes.fr/bench4star/

5. EXPERIMENTATIONS ET RESULTATS

Afin de tester notre approche proposée pour le contrôle de l'énergie, une série d'expériences a été établie. Dans la première expérience (**Table 0.3**), une solution initiale a été générée aléatoirement. Ensuite, cette solution a été comparée avec celle obtenue par l'algorithme génétique (NSGA-2).

	Solution initiale		***NSGA-2***		***Gain***		***Contrôle de perte d'énergie***	***Gain***
Ordres de production	**cmax (s)**	**énergie (s)**	**cmax (s)**	**énergie (s)**	**cmax %**	**énergie %**	**énergie (s)**	**énergie %**
1xAIP	486 s	68w	239 s	37.33w	50.9%	45.11%	33.3w	51.02%
2xAIP	707 s	98.98w	355 s	71.11w	49.78%	28.16%	66.2w	33.11%
3xAIP	873 s	136.45w	449 s	104 w	48.56%	23.8%	100w	26.80%
6xAIP	1592 s	234.37w	829 s	204 w	47.92%	12.96%	200w	14.66%
1xBELT	512 s	93.54w	348 s	57.77 w	32.00%	38.24%	53.33w	43.00%
1xBELT 1xAIP	860 s	121.25w	409 s	91.11 w	45.10%	25.02%	86.66w	28.52%
2xAIP 1xLATE	929 s	162.70w	509 s	124 w	45.20%	23.80%	120w	26.24%

Table 0.3 Efficience de l'énergie

Dans cette expérience, deux fonctions objectif ont été prises en considération : cmax (durée de production en secondes) et la consommation totale de l'énergie (watts). Le coefficient de chaque fonction est 0.5. Les solutions de différentes ordres de production obtenues par NSGA-2 sont nettement meilleures que les solutions initiales générées aléatoirement. Les deux dernières colonnes présentent la consommation et le gain de l'énergie, en utilisant le modèle de contrôle de perte de l'énergie (**Figure 0.5**).

Dans la deuxième expérience, nous testons la réactivité de l'algorithme proposé de contrôle de consommation de l'énergie. L'expérience s'applique sur l'ordre de production 2xAIP. Nous supposons dance cette expérience le scenario dans lequel à n'importe quel moment, un seuil de consommation de l'énergie est imposé. Comme cité précédemment, les causes de ce scenario sont multiples :

- baisse de l'énergie disponible (panneaux photo voltaïques, éoliennes, etc.).

- tarification dynamique de l'énergie, ce qui oblige le client à limiter sa consommation pour éviter un sur coût.

- un contrat entre le client et le serveur de l'énergie qui consiste à limiter le seuil de consommation de l'énergie

La **Figure 0.14** présente le diagramme de Gantt et un graphe de consommation de l'énergie sans un seuil d'énergie pour l'ordre de fabrication 2xAIP.

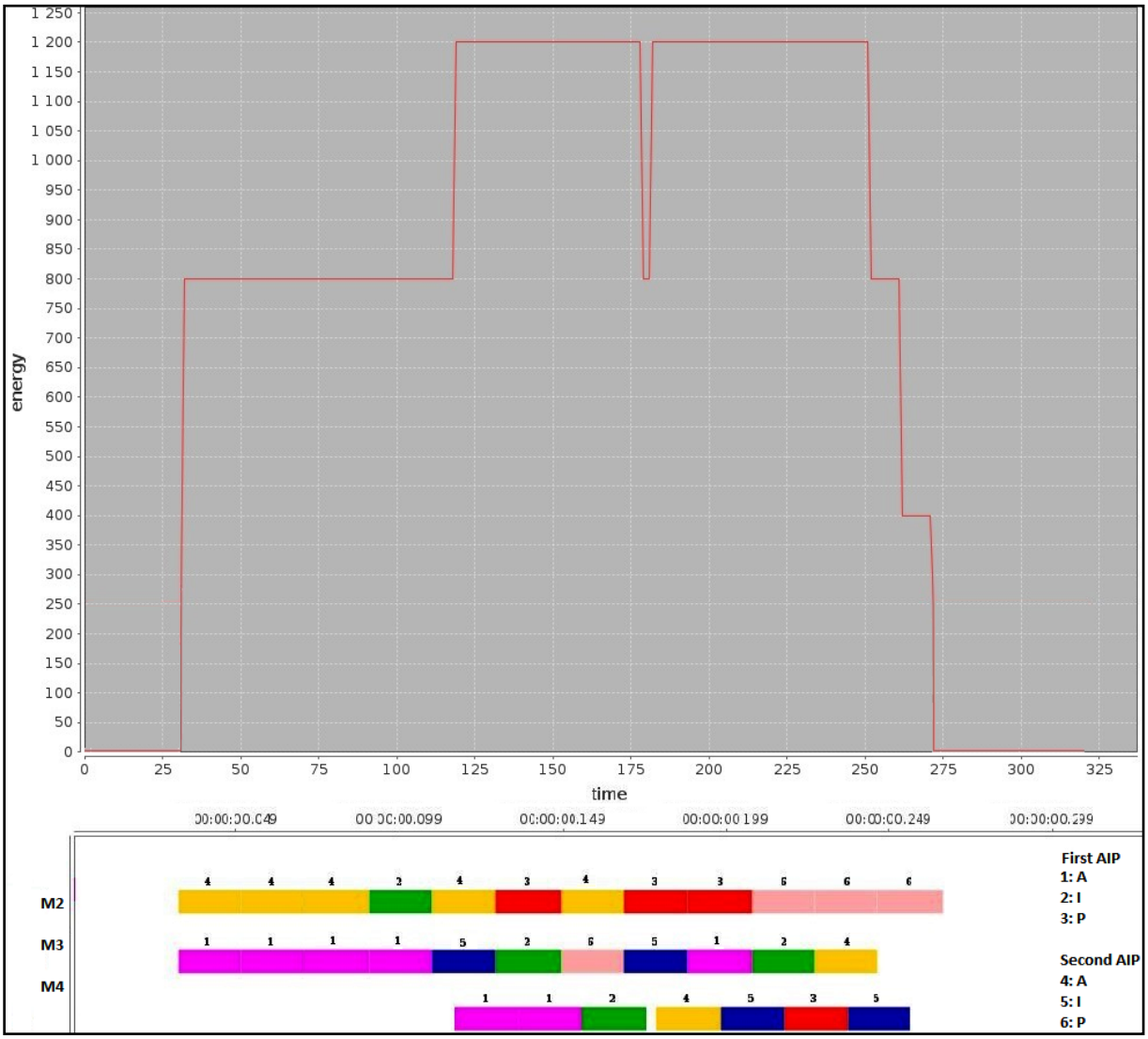


Figure 0.14 Consommation d'énergie et diagramme de Gantt sans seuil 2xAIP

Dans un premier temps, les trois machines attentent l'arrivé des convoyeurs transportant les produits, c'est ce qui explique que la consommation de l'énergie est nulle dans cette phase. Dés l'arrivée des produits, les machines commencent l'assemblage. Le pic de l'énergie remonte à 800 watts et atteint 1200 watts.

Dans la **Figure 0.15**, le seuil de consommation est fixé à 800 watts.

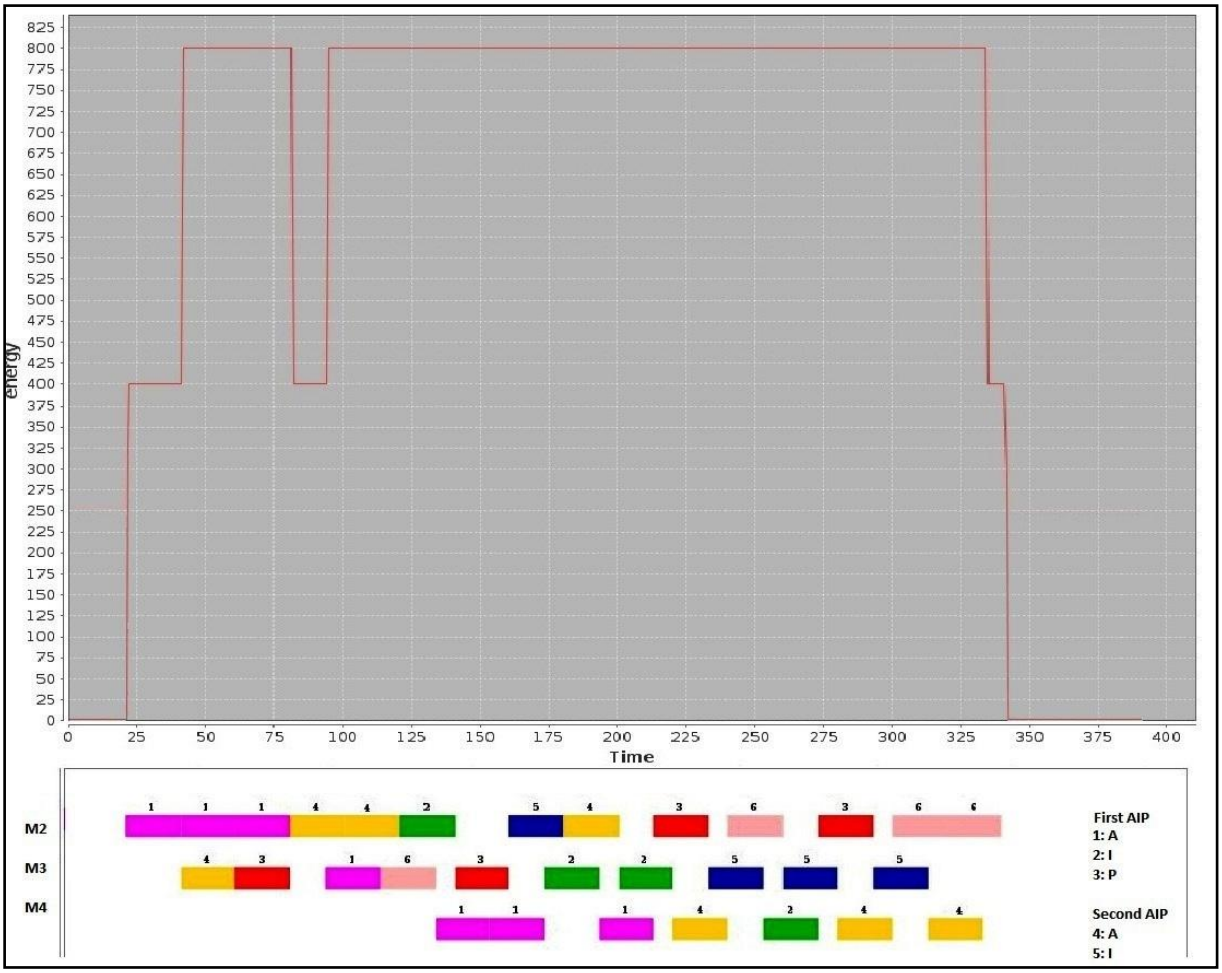


Figure 0.15 Consommation d'énergie et diagramme de Gantt avec un seuil = 800 watts

Après la première phase d'attente des produits, l'assemblage des produits est lancé par les trois machines (2,3 et 4). Nous constatons que l'algorithme de contrôle de pic de l'énergie empêche le dépassement du seuil imposé (seuil = 800 watts). Par contre, la durée de production est relativement longue par rapport à la première solution, dans laquelle le seuil de l'énergie est illimité.

Dans la **Figure 0.16**, le seuil de l'énergie est 400 watts. Nous constatons que le pic de l'énergie est stable durant l'assemblage des produits et la répartition des opérations sur les trois machines est relativement équilibrée. La durée de production de cette solution est plus importante par rapport aux deux premières solutions.

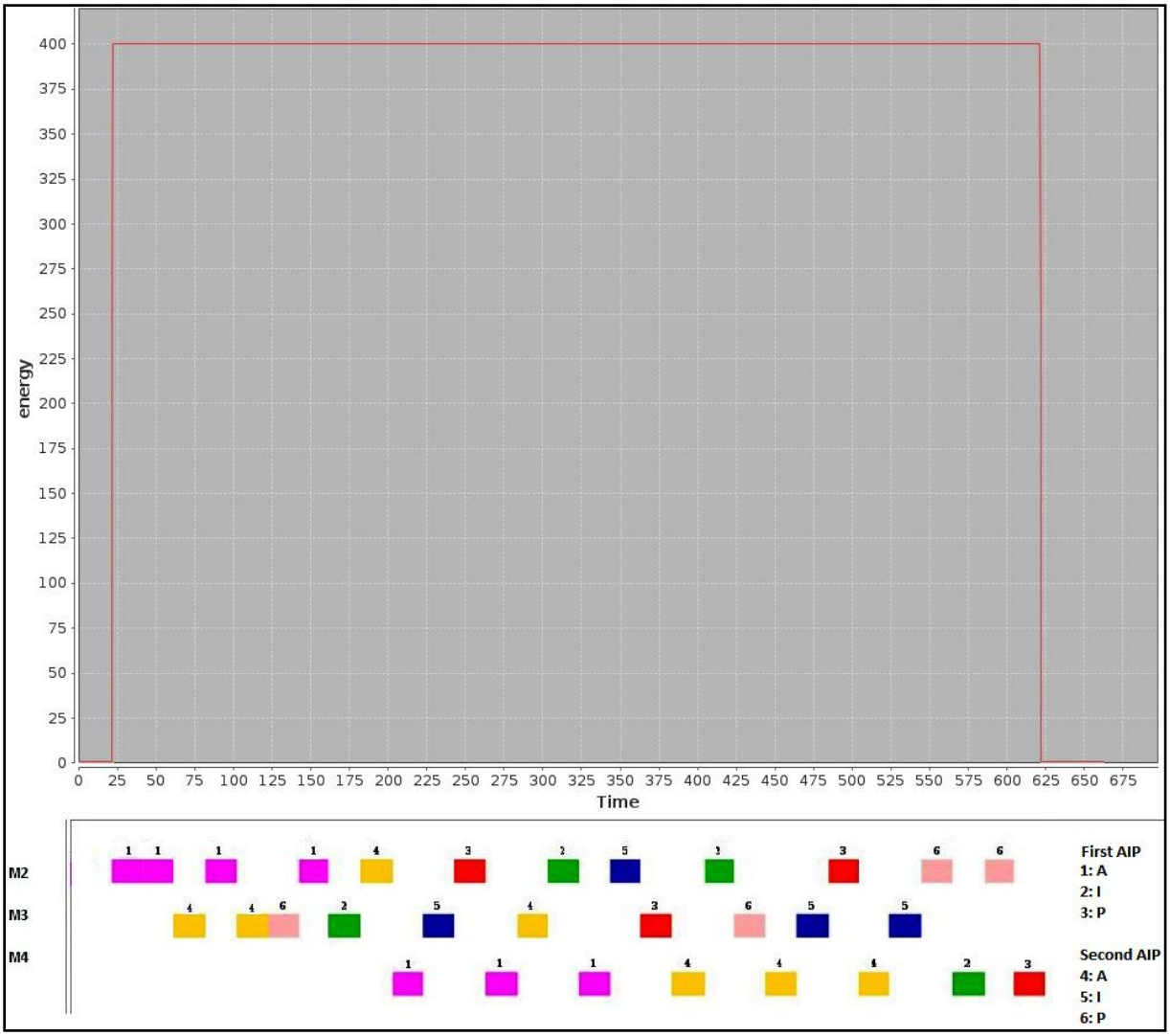


Figure 0.16 Consommation d'énergie et diagramme de Gantt avec un seuil = 400 watts

Dans cette expérience, nous testons la réactivité de l'approche lorsque le seuil est dynamique. Au départ, le seuil de l'énergie est fixé à 800 watt, ensuite il passe à 1200 watts. Dans une troisième phase, il chute à 0 watts et remonte en fin à 1200 watts. L'algorithme de contrôle de l'énergie réagit à chaque fois qu'un nouveau seuil est exigé.

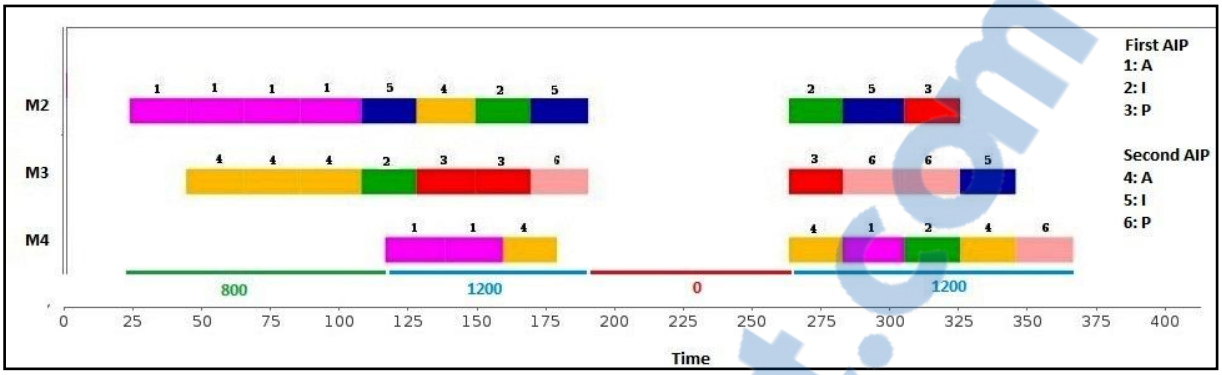


Figure 0.17 Consommation de l'énergie avec des seuils dynamiques

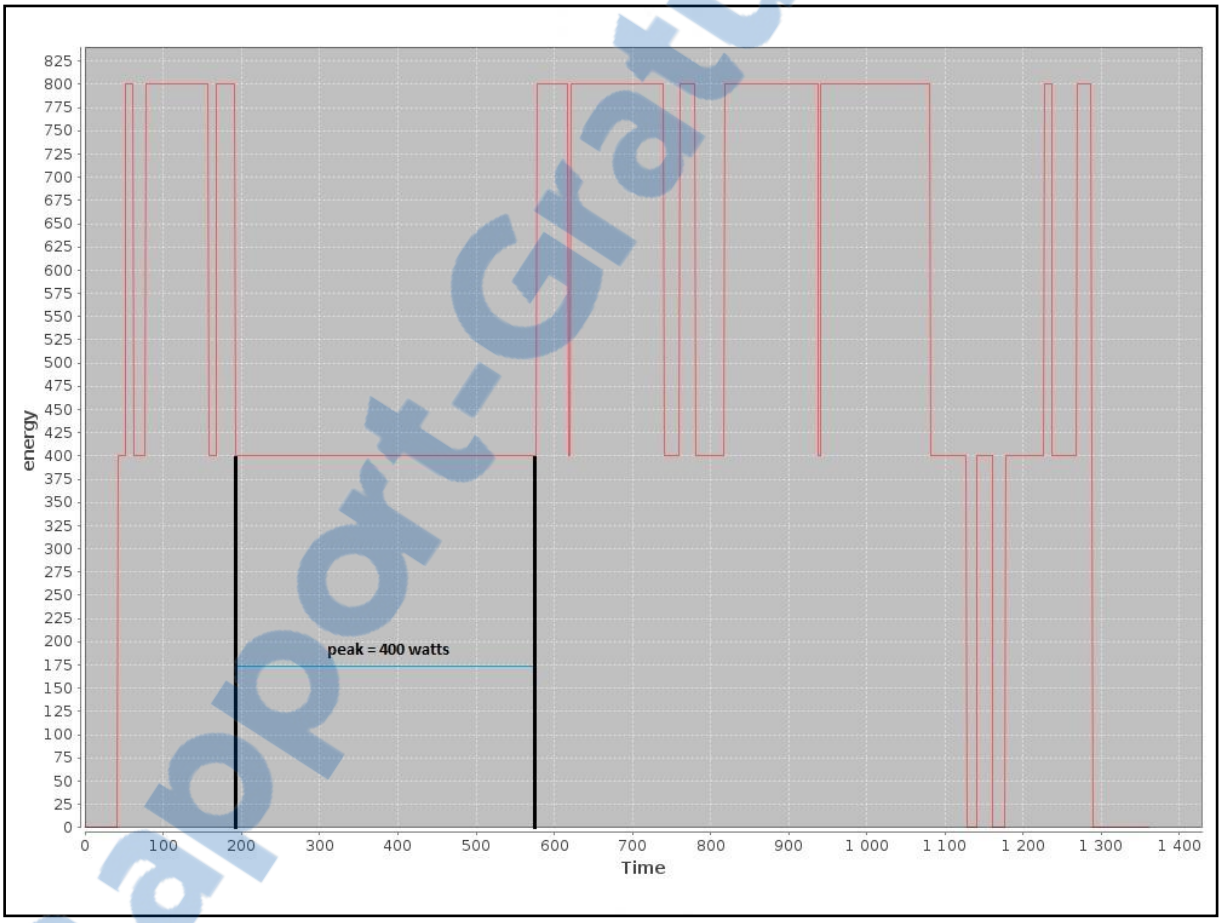


Figure 0.18 consommation de l'énergie avec un seuil =400 watts entre 199 et 570 secondes

La **Figure 0.18** présente un scénario dans lequel le seuil de l'énergie imposé est de 400 watts, mais dans l'intervalle 199 et 570 secondes.

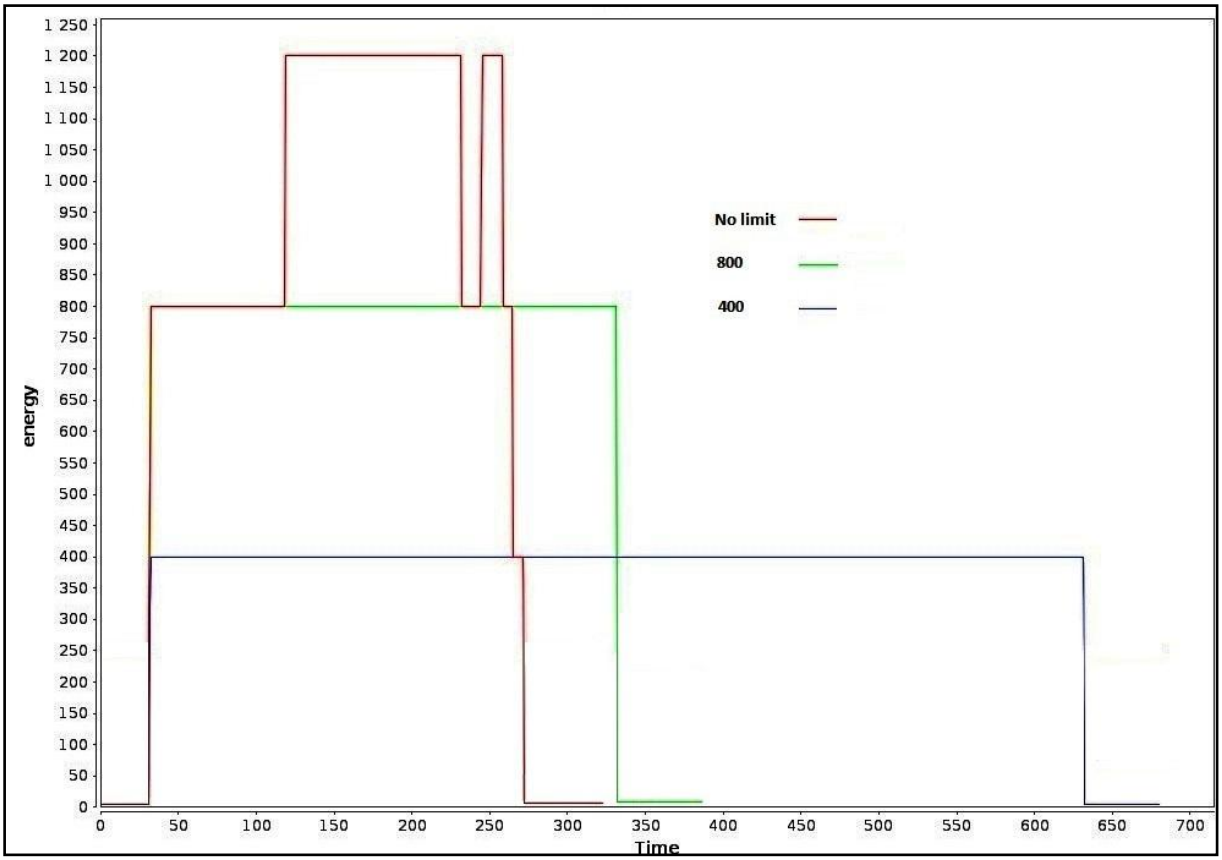


Figure 0.19 comparaison de trois scénarios de consommation de l'énergie (sans seuil, seuil = 800 watts, seuil = 400 watts)

(C. Pach, T. Berger, et al. 2015) ont proposé une approche réactive pour le contrôle de l'énergie, basée sur la technique des champs potentiels (*Potential Fields*). Ils ont appliqué cette approche sur la même cellule de production. Ses résultats obtenus, ont été comparés à nos résultats (**Table 0.4**).

		Champs potentiels	NSGA-2-ECA*
Seuil de l'énergie	**Ordre de production**	**Cmax (secondes)**	
800 watts	2XAIP	400 s	**340 s**
400 watts	2XAIP	650 s	**621 s**
Dynamique	2XAIP	520 s	**400 s**
	6XAIP	940 s	**844 s**

Table 0.4 comparaison entre NSGA-2ECA et champs potentiels (C. Pach, T. Berger, et al. 2015)

La **Table 0.4** présente une comparaison entre l'approche de (C. Pach, T. Berger, et al. 2015) et notre approche (NSGA-2-ECA : Non-dominant Sorting Genetic Algorithm-Energy Control Algorithm). Le temps de production est réduit à 15 % pour un seuil de 800 watts et à 5.4 % pour un seuil de 400 watts. Pour les seuils dynamiques, le temps de production est réduit à 23.07% pour 2XAIP et 10.21% pour 6XAIP.

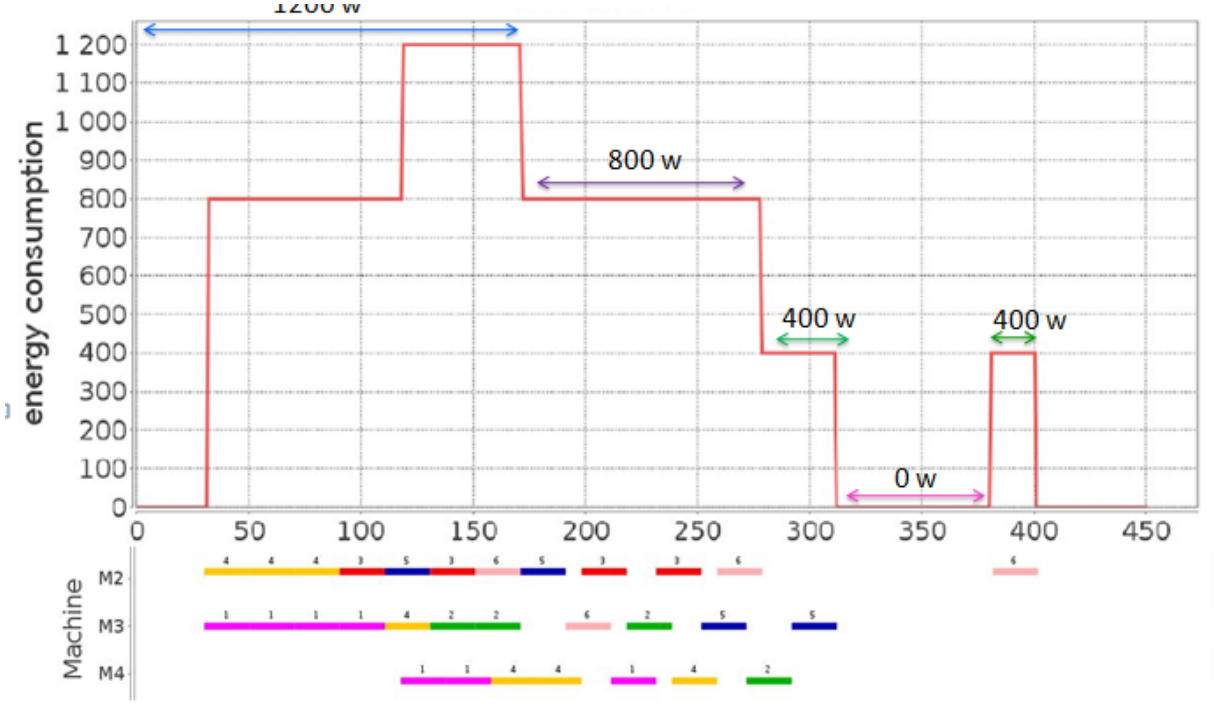


Figure 0.200 Meilleure solution avec des seuils dynamiques pour l'ordre de production 2XAIP

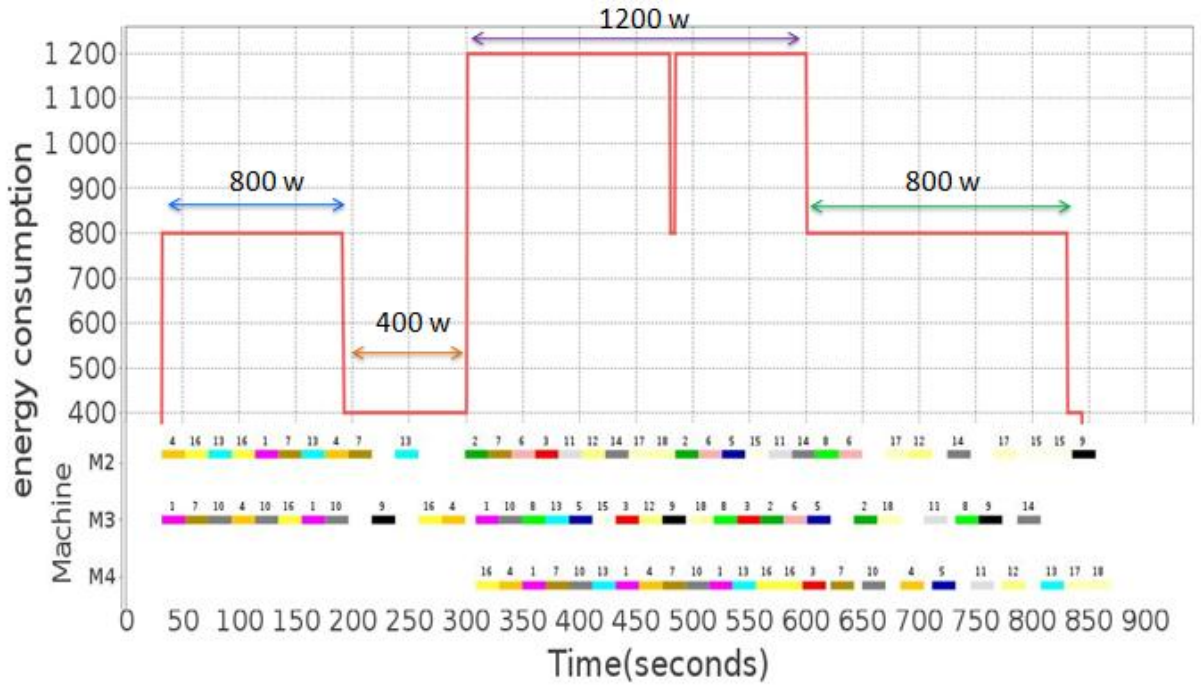


Figure 0.21 Meilleure solution avec des seuils dynamiques pour l'ordre de production 6XAIP

6. CONCLUSION

Nous avons proposé dans ce chapitre, une approche prédictive-réactive pour le contrôle de consommation de l'énergie. L'algorithme de contrôle de pic de consommation de l'énergie permet de recalculer l'ordonnancement initial afin de limiter le pic de consommation de l'énergie.

L'algorithme génétique NSGA-2 peut être utilisé comme une méthode prédictive en cas d'une nécessité d'une baisse de pic de l'énergie. Trois critères sont pris en considération : la durée de production, consommation de l'énergie et le pic de consommation de l'énergie.

Les résultats obtenus et comparé avec les résultats de (C. Pach, T. Berger, et al. 2015) montrent que l'algorithme génétique est utile pour anticiper une baisse de l'énergie.

Nous avons proposé aussi un modèle pour limiter la perte de l'énergie lorsque les machines sont en attente de l'arrivée des convoyeurs transportant les productions à assembler. Cette technique consiste à arrêter les machines lorsqu'elles sont en attente et les démarrer en cas d'arrivé d'un produit. Mais, cette technique peut causer des dommages matériels.

D'après ces expériences, l'ordonnancement peut jouer un rôle crucial dans la consommation de l'énergie. Une méthode d'ordonnancement qui prend en considération la consommation de

l'énergie peut limiter l'émission du CO2, ce qui apporte un gain sur le plan économique et écologique.

CONCLUSION GENERALE

Nous avons présenté dans cette thèse deux méthodes d'ordonnancement destinées aux systèmes manufacturiers de type job shop flexible avec la prise en considération du temps de transport entre les machines.

Malgré la grande variété des méthodes proposées, il n'existe aucune méthode qui peut atteindre l'optimalité avec certitude. Donc, ce problème reste encore une piste ouverte aux chercheurs.

La première méthode proposée a pour but de minimiser la durée de production (cmax). Un algorithme génétique amélioré par une recherche tabou, en proposant une nouvelle fonction de voisinage. Cette fonction de voisinage est basée sur deux algorithmes, exploitant les fenêtres temporelles libres et occupées.

Nous avons considéré dans la deuxième méthode la consommation de l'énergie durant le processus de production. Par conséquent, le problème d'optimisation devient un problème multi-objectifs. La particularité de notre méthode réside dans l'anticipation d'une baisse d'énergie. Pour ce faire, nous avons proposé un algorithme génétique (NSGA-2), en utilisant trois fonctions objectifs : la durée de production, la consommation totale de l'énergie et le pic de l'énergie consommée. Ensuite, une seule solution est sélectionnée parmi l'ensemble de solutions non dominées par une technique, nommée AHP (Analytic Hierarchy Process (Saaty 2008). Ensuite, l'algorithme Energy Control proposé, contrôle le pic de consommation de l'énergie d'une manière réactive. Les résultats obtenus sur la cellule de production AIP-PRIMCA ont été comparés avec une étude (C. Pach, T. Berger, et al. 2015) dans laquelle la cellule de production AIP-PRIMECA a été utilisée. Apres une réaction face à une exigence de baisse du pic d'énergie consommée, le temps de production obtenu par notre méthode prédictive-réactive est plus court par rapport à la méthode réactive, basée sur les champs potentiels, proposé par (C. Pach, T. Berger, et al. 2015).

Cette exigence de baisse de l'énergie est causée par plusieurs facteurs : baisse d'énergie stockée, après une perturbation au niveau de réapprovisionnement d'énergie (surtout lors de l'utilisation des nouvelles sources d'énergie telle que l'énergie solaire, éolienne, etc. En plus, parfois, un contrat entre le client et le serveur de l'énergie qui exige au client de ne pas dépasser un certain seuil de consommation. La tarification de prix d'énergie peut être dynamique, d'où la nécessité de s'adapter afin d'éviter une éventuelle pénalité

PERSPECTIVES

Les expériences réalisées donnent lieu à plusieurs perspectives :

- Ajouter d'autres critères pour l'optimisation, tel que les émissions du co2.
- Prendre en considération d'autres scénarios imprévus, tel que les pannes des machines, des nouveaux ordres de productions, changement de priorité des ordres de production.
- Optimiser le fonctionnement du switcher afin de ne pas endommager les machines.
- La meilleure façon de traiter le problème de l'énergie est de travailler en groupes de plusieurs spécialités: électronique, mécanique, informatique, conception des produits, etc. Donc, ca serait mieux de travailler en collaboration avec d'autres spécialistes pour de meilleures performances.

Résumé :

Nous proposons deux méthodes d'ordonnancement, destinées aux ateliers de production de type job shop flexible, en prenant en considération le transport entre machines. Un algorithme génétique amélioré par une recherche tabou avec une nouvelle fonction de voisinage pour minimiser la durée de production. Nous proposons ensuite, une méthode prédictive-réactive pour contrôler l'énergie consommée durant la production. Un algorithme génétique multi-objectifs (NSGA-2) avec trois objectifs (durée de production, consommation totale de l'énergie et le pic de consommation de l'énergie) est utilisé comme méthode prédictive. Un algorithme réactif de contrôle du pic de l'énergie consommée est proposé pour recalculer l'ordonnancement face à une exigence de baisse de consommation d'énergie. L'approche a été testée sur des benchmarks de la cellule de production AIP-PRIMECA, développée à l'université de Valenciennes en France. Les résultats ont été comparés avec d'autres études sur AIP-PREMICA.

Mots-clés *:*

Systèmes manufacturiers, optimisation mono objectif, optimisation multi-objectifs, contrôle d'énergie, méthode prédictive-réactive, NSGA-2, Ordonnancement du Job-shop flexible.

Abstract

We propose two methods for flexible job shop scheduling problem with transport time between machines. A genetic algorithm improved by a Taboo search with a new neighborhood function, developed to minimize the overall production time. In the second method, a predictive-reactive method is developed to control the energy consumption during manufacturing operations. A multi-objective genetic algorithm (NSGA-2) is proposed, with three objective functions (overall production time, overall energy consumption and the peak of energy consumption). This NSGA-2 is used as predictive method. We propose a reactive algorithm to control the peak of energy consumption to recalculate the initial schedule against new exigencies of power peak consumption.

The two methods were tested on AIP-PRIMECA which reflects a real flexible job shop with transport time between machines. The results have been compared to other studies on AIP-PREMICA.

Keys words

Manufacturing systems, mono-objective optimization, multi-objective optimization, energy control, predictive-reactive approach, NSGA-2, flexible job shop scheduling problem.

ملخص

نقترح طريقتين للورشات الصناعية التي تعرف بإسم Flexible Job shop Scheduling مع وقت النقل بين الآلات. تم تطوير خوارزمية وراثية لتقليل وقت الإنتاج الكلي، تم تحسينها بواسطة تقنية Taboo Searchمع اقتراح خوارزمية لإيجاد الحلول الجوارية (neighborhood function) . في الطريقة الثانية ، تم تطوير طريقة تفاعلية للتحكم في استهلاك الطاقة أثناء عمليات التصنيع. يتم اقتراح خوارزمية جينية متعددة الأهداف (NSGA-2) ، مع ثلاث أهداف (إجمالي وقت الإنتاج ، واستهلاك الطاقة الإجمالي وذروة استهلاك الطاقة). تستخدم الخوارزمية NSGA-2 كطريقة تنبؤية. نقترح خوارزمية تفاعلية للتحكم في ذروة استهلاك الطاقة لإعادة حساب الجدول الأولي(Initial Scheduling) مقابل المتطلبات الجديدة المتعلقة بذروة استهلاك الطاقة.

تم اختبار الطريقتين على AIP-PRIMECA الذي التي تعكس ورشة صناعية حقيقية Flexible Job shop Scheduling مع وقت النقل بين الآلات. تم مقارنة نتائجنا مع نتائج لدراسات أخرى على AIP-PREMICA.

كلمات المفاتيح

نظم التصنيع ، التحسين الأحادي الموضوع ، التحسين المتعدد الأهداف ، التحكم في الطاقة ، النهج التنبؤي التفاعلي ، الخوارزمية الجينية NSGA-2 ، مشكلة جدولة الوظائف المرنة FJS.

1. BIBLIOGRAPHIE

Afsar, H. Murat, Philippe Lacomme, Libo Ren, et Daniele Vigo. «Resolution of a Job-Shop problem with transportation constraints: a master/slave approach.» *IFAC-PapersOnline 49-12*. Elsevier, 2016. 898-903.

Aydin, Hakan, Rami Melhem, Daniel Mosse, et Pedro Mejia-Alvarez. «Optimal reward-based scheduling of periodic real-time tasks.» Édité par IEEE. *IEEE Transactions on Computers* 50, n° 2 (1999): 111 - 130.

Barichard, Vincent. «Une approche hybride pour l'optimisation.» Thèse de doctorat, Université d'Angers, France, 2003.

Barnier, Nicolas, et Pascal Brisset. «Optimisation par algorithme génétique sous contraintes.» *Techniques et sciences informatiques* 18, n° 2 (1999): 137-159.

Beasley, J.E. «OR-library: distributing test problems by electronic mail.» *Journal of Operational Research Society* 41, n° 11 (1990): 1069-1072.

Bekkar, Azzedine, Oualid Guemri, Abdelghani Bekrar, Nassima Aissani, Bouziane Beldjilali, et Damien Trentesaux. «An Iterative Greedy Insertion Technique for Flexible Job Shop Scheduling Problem.» *8th IFAC Conference on Manufacturing Modelling, Management and Control*. Troyes, France, 2016. 1956–1961.

Bertel, Sylvain. *Les problèmes d'ordonnancement de type flow-shop hybride avec recirculation et fênetres de temps.* Thèse de doctorat, Tours, France: Université de Tours, 2001.

Blondel, françois. *Gestion de la production.* 3 eme édition. Paris: Dunod, 2004.

Boukef, Hela. «Sur l'ordonnancement d'ateliers job-shop flexibles et flow-shop en industries.» Thèse de doctorat, Ecole Centrale de Lille, 2009.

Brandimarte, Paolo. «Routing and scheduling in a flexible job shop by tabu search.» *Annals of Operations Research, SpringerLink* 41, n° 3 (1993): 157–18.

Brucker, Peter, Silvia Heitmann, et Peter Brucker. «Flow-shop problems with intermediate buffers.» *Operations Research-Spektrum* 25, n° 4 (2003): 549-574.

Bruzzone, A, D Anghinolfi, M Paolluci, et F Ronelli. «Energy-aware scheduling for improving manufacturing process sustainability: A mathematical model for flexible flow shops.» Édité par ELSEVIER. *CIRP Annals* 61, n° 1 (2012): 459-462.

Bürgy, Reinhard, et Heinz Gröflin. «The blocking job shop with rail-bound transportation.» *Journal of Combinatorial Optimization* 31, n° 1 (2016): 152–181.

Carlier, François. *Problèmes d'ordonnancement : modélisation, complexité algorithmes.* Édité par Dunod. 1997.

Carlier, Jacques. «Problème d'ordonnancement à contraintes de ressources: algorithmes et complexité.» Thèse de doctorat, Université de Paris VI, Paris, France, 1984.

Caumond, A., P. Lacomme, A. Moukrim, et N. Tchernev. «An MILP for scheduling problems in an FMS with one vehicle.» *European Journal of Operational Research* 199, n° 3: 706-722.

Chaari, Tarek. «Un algorithme genetique pour l'ordonnancement robuste: application au probleme flow shop hybride.» Thèse de doctorat, Université de Valenciennes et du Hainaut Cambrésis, France, Valenciennes, France, 2010.

Charon, Irène. *Méthodes d'optimisation combinatoire.* 1er édition. Elsevier-Masson, 1996.

Chrétienne, Pascal. «Les réseaux de petri temporises.» Thèse de doctorat, Paris, France, 1983, Université de Paris VI.

Christian, Artigues, Briand Cyril, Marie-Claude Portmann, et François Roubellat. *Pilotage d'atelier basé sur un ordonnancement flexible.* Chapitre d'ouvrage, Hermes Lavoisier, 2002, 61-97.

Collette, Yann, et Patrick Siarry. *Optimisation multiobjectif.* 1ère édition. Eyrolles, 2002.

Corne, D.W. «The Pareto envelope based selection algorithm for multi-objective optimization.» *Lecture Notes in Computer Science*, 2000.

Crama, Y, V Kats, J van de Klundert, et E Levner. «Cyclic scheduling in robotic flowshops.» *Annals of Operations Research* 96, n° 1 (2000): 97–124.

Dai, Min, Dunbing Tang, Adriana Giret, Miguel Salido, et W.D. Li. «Energy-efficient scheduling for a flexible flow shop using an improved genetic-simulated annealing algorithm.» *Robotics and Computer-Integrated Manufacturing* 29, n° 5 (2013): 418–429.

Deb, Kalyanmoy, Amrit Pratap, Sameer Agarwal, et T Meyarivan. «A fast and elitist multi-objective genetic algorithm: NSGA-2.» *IEEE Transactions on Evolutionary Computation* 6, n° 2 (2002): 182-197.

Drakaki, Maria, et Panagiotis Tzionas. «Manufacturing Scheduling Using Colored Petri Nets and Reinforcement Learning.» *Applied Sciences* 7, n° 2 (2017): 2-3.

Duvivier, David. «Étude de l'hybridation des méta-heuristiques, application à un problème d'ordonnancement de type jobshop.» Thèse de doctorat, Université du Littoral Côte d'Opale, Dunkerque, France, 2000.

Esquirol, Patrick, et Pierre Lopez. *L'ordonnancement.* 1er édition. Economica, 1999.

Fan, Jiyuan, et Stuart Borlase. «The Evolution of Distribution.» Édité par IEEE. *IEEE Power and Energy Magazine* 7, n° 2 (2009): 63-68.

Fang, Kan, Nelson A. Uhan, Fu Zhao, et John W. Sutherland. «Flow shop scheduling with peak power consumption constraints.» *Annals of Operations Research* 206, n° 1 (2013): 115–145.

Fang, Kan, Nelson Uhan, Fu Zhao, et John W Sutherland. «A new approach to scheduling in manufacturing in power consumption for power consumption and carbon footprint reduction.» *Journal Manufacturing System* 30, n° 4 (2011): 234-240.

Fonseca, Carlos, et Peter Fleming. «An overview of evolutionary algorithms in multiobjective optimization.» *Evolutionary Computation* 3, n° 1 (1995): 1-16.

Fontanili, Franck. «Integration d'outils de simulation et d'optimisation pour le pilotage d'une ligne d'assemblage multiproduit a transfert asynchrone.» Thèse de doctorat, Université de Paris XIII, 1999.

Fred, Glover. «Tabu search.» *ORSA, Journal of Computing* 1, n° 2 (1989): 190-206.

Garey, Michael R., et David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness (Series of Books in the Mathematical Sciences).* 1st edition. W. H. Freeman and Company, 1979.

Gargouri, Emna. «Ordonnancement coopératif en industrie agroalimentaire.» Thèse de doctorat, Université Lille1 - Sciences et Technologies, Lille, France, 2003.

Ghadimi, Pouya, Sami Kara, et Bernard Kornfeld. «Renewable energy integration into factories: Real-time control of on-site energy systems.» *CIRP Annals - Manufacturing Technology* 64, n° 1 (2015): 443–446.

Giard, Vincent. *Gestion de la production.* 3e édition. Paris: Economica, 2003.

Goldberg, David. *Algorithmes génétiques exploration, optimisation et apprentissage automatique.* 1st edition. Dorling Kindersley Pvt Ltd, 2008.

Gotha. «scheduling problems.» *Operations Research* 27, n° 1 (1993): 77-150.

Hao, Jin-Kao, Philippe Galinier, et Michel Habib. «Métaheuristiques pour l'optimisation combinatoire et l'affectation sous contraintes.» *Revue d'Intelligence Artificielle* (Lavoisier) 13, n° 2 (1999): 283-324.

Haupt, Randy L, et Sue Ellen Haupt. *Practical Genetic Algorithms.* 2nd edition. Wiley, 2004.

He, Yan. «Job Scheduling Model of Machining System for Green Manufacturing.» *Journal of Mechanical Engineering* 43, n° 4 (2007): 27-33.

Herrmann, Jeffrey W. *Handbook of Production Scheduling.* Vol. 89. Springer Science & Business Media, 2006.

Huang, Rong-Hwa, Shun-Chi Yu, et Po-Han Chen. «Energy-Saving Scheduling in a Flexible Flow Shop Using a Hybrid Genetic Algorithm.» *Earth & Environmental Sciences* 8, n° 10 (2017): 1037-1056.

Hurink, Johann, et Sigrid Knust. «A tabu search algorithm for scheduling a single robot in a job-shop environment.» *Discrete Applied Mathematics* 119, n° 1 (2002): 181-203.

Hurink, Johann, et Sigrid Knust. «Tabu search algorithms for job-shop problems with a single transport robot.» *European Journal of Operational Research* 162, n° 1 (2005): 99-111.

Javel, Georges. *Organisation et gestion de la production.* 3e édition. DUNOD, 2004.

Jiang, Zengqiang, et Le Zuo. «Study on multi-objective flexible job-shop scheduling problem considering energy consumption.» *Journal of Industrial Engineering and Man* 7, n° 3 (2014): 589-604.

Jones, Albert, Luis Rabelo, et Abeer T Sharawi. «Survey of Job Shop Scheduling Techniques.» Dans *Wiley Encyclopedia of Electrical and Electronics Engineering*. 1998.

Jouglet, Antoine, Philippe Baptiste, et Jacques Carlier. «Exact procedures for single machine total cost scheduling.» Édité par IEEE. *IEEE International Conference on Systems, Man and Cybernetics.* Yasmine Hammamet, Tunisia, Tunisia, 2002.

Kacem, Imed, Slim Hammadi, et Pierre Borne. «Approach by localization and multiobjective evolutionary optimization for flexible job-shop scheduling problems.» *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* (IEEE) 32, n° 1 (2002): 1 - 13.

Khatib, Oussama. «The Potential Field Approach And Operational Space Formulation In Robot Control.» Dans *Adaptive and Learning Systems*, 367-377. Stanford University, USA: Springer US, 1986.

King, Russell E, Thom J. Hodgson, et Franklin W Chafee. «ROBOT TASK SCHEDULING IN A FLEXIBLE MANUFACTURING CELL.» *IIE Transactions* 25, n° 2 (1993): 80-87.

Kirkpatrick, Scott. «Optimization by simulated annealing: Quantitative studies.» *Journal of Statistical Physics* 34, n° 5 (1984): 975–986.

Knowles, Joshua W, et David W Corne. «The Pareto archived startegy: a new baseline algorithm for multiobjective optimization.» *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*, 1999: 98-105.

Knust, S. «Shop-Scheduling Problems with Transportation.» Ph. D. Thesis, Fachbereich Mathematik /Informatik Universität Osnabrück, 1999.

Kuhlmann, Timm, et Thomas Bauernhansl. «Method for Designing an Energy-agile Energy System for Industrial Manufacturing.» *The 22nd CIRP Conference on Life Cycle Engineering* 29 (2015): 179-184.

Lacomme, Philippe, Christian Prins, et Marc Sevaux. *Algorithmes de graphes.* 2ème édition. Eyrolles, 2003.

Lacomme, Philippe, et Mohand Said Larabi. «A System Generation Schedules for transportation problem in job-shop environment.» *COSI 2008.* Tizi-Ouzou Algérie, 2008.

Lacomme, Philippe, Mohand Larabi, et Nikolay Tchernev. «Job-shop based framework for simultaneous scheduling of machines and automated guided vehicles.» *International Journal of Production Economics* 143, n° 1 (2013): 24-34.

Lacomme, Philippe, Mohand Said Larabi, et Nikolay Tchernev. «A disjunctive graph for the job- shop with several robots.» *MISTA conference.* Paris, France, 2007.

Larabi, Mohand. «Le problème de job-shop avec transport : modélisation et optimisation.» Thèse de doctorat, Université Blaise Pascal - Clermont-Ferrand II, 2010.

Lawrence, Davis. «Job Shop Scheduling with Genetic Algorithms.» *Proceedings of the 1st International Conference on Genetic Algorithms.* ACM Digital Library, 1985. 136-140.

Lee, K.-M, T Yamakawa, et Keon-Myung Lee. «A genetic algorithm for general machine scheduling problems.» *Knowledge-Based Intelligent Electronic Systems, 1998. Proceedings KES '98. 1998 Second International Conference on.* Adelaide, SA, Australia, Australia: IEEE, 1998. 60-66.

Letouzey, Agnès. «Ordonnancement interactif basé sur des indicateurs : Applications à la gestion de commandes incertaines et à l'affectation des opérateurs.» Thèse de doctorat, Institut National Polytechnique de Toulouse, Toulouse, France, 2001.

Liouane, Noureddine. «Contribution à l'élaboration d'un outil d'aide à la décision pour l'ordonnancement de production en environnement incertain.» Thèse de doctorat, Université Lille1 - Sciences et Technologies, Lille, 1998.

Luo, Hao, Bing Du, George Huang, Huaping Chen, et Xiaolin Li. «Hybrid flow shop scheduling considering machine electricity consumption cost.» *International Journal of Production Economics* 146, n° 2 (2013): 423–439.

Marik, Vladimir, et Duncan McFarlane. «Indutrial adoption of agent-based technologies.» *IEEE Inteligent Systems* 20, n° 1 (2005): 27-35.

Mastrolilli, Monaldo, et Luca Maria Gambardella. «Effective neighbourhood functions for the flexible job shop problem.» *Journal of Scheduling* 3, n° 1 (2000): 3-20.

Mesghouni, Khaled. «Application des algoriathmes évolutionnistes dans les problèmes d'optimisation en ordonnancement de la production.» Thèse de doctorat, Université de sciences et technologies de Lille 1, Lille, France, 1999.

Moslehi, Ghasem, et Mehdi Mahnam. «A Pareto approach to multi-objective flexible job shop scheduling problem using particle swarm and local search.» *International Journal of Production Economics* 129, n° 1 (2011): 14-22.

Nakano, Ryohei, et Takeshi Yamada. «Conventional genetic algorithm for job shop problems.» *Proceedings of the fourth International Conference on Genetic Algorithms.* San Diego, Calofornia, 1991. 474 -479.

Nowicki, Eugeniusz. «The permutation flow shop with buffers: A tabu search approach.» *European Journal of Operational Research* 116, n° 1 (1999): 205-219.

Pach, C, T Berger, Y Sallez, et Trentesaux. «Reactive control of overall power consumption in flexible manufacturing systems scheduling: A Potential Fields model.» *Control Engineering Practices*, 2015: 44, 193-208, www.elsevier.com/locate/conengprac.

Pach, C, T Berger, Y Sallez, T Bonte, et E Adam. «Reactive and energy-aware scheduling of flexible manufacturing systems using potential fields.» *Computers in Industry*, 2014: 65 (3), 434–448.

Pach, Cyrille, Thierry Berger, Thérèse Bonte, et Damien Trentesaux. «ORCA-FMS: a dynamic architecture for the optimized and reactive control of flexible manufacturing scheduling.» Édité par Elsevier. *Computers in Industry* 65, n° 4 (2014): 706-720.

Paschos, Vangelis. *Optimisation combinatoire 1: concepts fondamentaux.* I vols. Hermes Science Publications, 2005.

Paulli, Jan. «A hierarchical approach for the FMS scheduling problem.» *European Journal of Operational Research* 86, n° 1 (1995): 32-42.

Peter Brucker, Bernd Jurisch & Bernd Sievers. «A branch and bound algorithm for the job-shop scheduling problem.» *Discrete Applied Mathematics* 49, n° 1 (1994): 107-127.

Pinedo, Michael. *Scheduling: Theory, Algorithms, and Systems.* 3rd edition. Springer Publishing Company, 1998.

Pouya, Ghadimi, Kara Sami, et Kornfeld Bernard. «Renewable energy integration into factories: Real-time control of on-site energy systems.» *CIRP Annals - Manufacturing Technology* 64, n° 1 (2015): 443–446.

Prabhu, v, d Trentesaux, et M Taisch. «Energy-aware manufacturing operations.» *International Journal of Production Research* 53, n° 23 (2015): 1-11.

Prabhu, V, et M Taisch. «Simulation modeling of energy dynamics in discrete manufacturing systems.» *Proceeding of the Information Control Problems in Manufacturing*, 2012: 740-745.

Prabhu, Vittaldas. «Services for Competitive and Sustainable Manufacturing in the Smart Grid in Service Orientation in Holonic and Multi-Agent Manufacturing Control.» Dans *Service Orientation in Holonic and Multi-Agent Manufacturing Control* , édité par Springer-Verlag Berlin, 227-240. 2012.

Pušavec, Franci, Peter Krajnik, et Janez Kopac. «Transitioning to sustainable production-Part I: application on machining technologies.» *Journal of Cleaner Production* 18, n° 2 (2010): 174-184.

Raich, Anagha, P J Nikumbh, et Manisha Patil. «Flexible Job–Shop Scheduling Problem Using Simulated Annealing.» *International Journal of Research & Reviewes in Computer Science* 3, n° 2 (2012): 1491-1501.

Raja, P, et S Pugazhenthi. «Path planning for a mobile robot in dynamic.» *International Journal of the Physical Sciences* 6, n° 20 (2011): 4721-4731.

Roghanian, Parastoo, Amran Rasli, et Hamed Gheysari. «Productivity Through Effectiveness and Efficiency in the Banking Industry.» *Procedia - Social and Behavioral Sciences* 40 (2012): 550-556.

Roy, Bernard. *Algebre moderne et theorie des graphes, orientees vers les sciences economiques et sociales.* 1er édition. Édité par Dunod. Vol. 1. 1969.

Saaty, Thomas L. «Decision making with the analytic hierarchy process.» *Int. J. Services Sciences* 1, n° 1 (2008): 83-98.

Sandhya, Pasala, Balla Nandana Kumar, et Suresh Chandra Satapathy. «A Study of Roulette Wheel and Elite Selection on GA to Solve Job Shop Scheduling.» Édité par Berlin, Heidelberg Springer. 2013. 477-485.

Schwiegelshohn, Uwe. «Preemptive Weighted Completion Time Scheduling of Parallel Jobs.» *SIAM Journal on Computing* 33, n° 6 (2012): 1280–1308.

Shrouf, Fadi, Joaquin Ordieres-Meré, Alvaro García-Sánchez, et Miguel Ortega-Mier. «Optimizing the production scheduling of a single machine to minimize total energy consumption costs.» *Journal of Cleaner Production* 67 (2014): 197–207.

Sridhar, N, M Victor Raj, et K Chandra sekar. «Minimizing Manufacturing Cost In Flexible Job-ShopScheduling Problems.» *International Journal of Artificial Intelligence and Mechatronics* 1, n° 5 (2013): 2320-5121.

Srinivas, N, et Kalyanmoy Deb. «Multi-objective function optimization using non-dominated sorting genetic algorithm.» *Evolutionary Computation* 2, n° 3 (1995): 221-248.

Strusevich, V.A. «A heuristic for the two-machine open-shop scheduling problem with transportation times.» *Discrete Applied Mathematics* 93, n° 1 (1999): 287-304.

Taillard, E. «Benchmarks for basic scheduling problem.» *European Journal Of Operational Research* 64, n° 2 (1993): 278-285.

Talb, El-Ghazali. *Metaheuristics: From Design to Implementation.* John Wiley and Sons, Hoboken, : 978-0-470-27858-1, New Jersey, USA, 2009.

Tangour, Fatma. «Ordonnancement dynamique dans les indutries agroalimentaires.» Thèse de doctorat, Ecole centrale de Lille, Lille, Ecole centrale de Lille, 2007.

Tarek, Chaari, Chaabane Sondes, Aissani Nassima, et Trentesaux Damien. «Scheduling under uncertainty: Survey and research directions.» *Conference: 3rd International Conference on Advanced Logistics and Transport, IEEE, At Hammamet, Tunisia*, 2014: DOI: 10.1109/ICAdLT.2014.6866316.

T'Kindt, Vincent, et Jean-Charles Billaut. *Multicriteria Scheduling: Theory, Models and Algorithms.* 2ed edition. Springer-Verlag Berlin Heidelberg, 2006.

Tonelli, Flavio, Alessandro A.G Bruzzone, Davide Anghinolfi, et Alessandro A.G Bruzzone. «Energy-aware scheduling for improving manufacturing process sustainability: A mathematical model for flexible flow shops.» Édité par ELSEVIER. *CIRP Annals* 61, n° 1 (2012): 459-462.

Trentesaux, d, et V Prabhu. «Sustainability in manufacturing operations scheduling: stakes, approaches and trends.» Dans *Advances in Production Management Systems. Innovative and Knowledge-Based Production Management in a Global-Local World*, 106-113. Springer, Berlin, Heidelberg, 2014.

Trentesaux, Damien, et al. «Benchmarking flexible job-shop scheduling and control systems.» *Control Engineering Practice* 21, n° 9 (2013): 1204–1225.

Vacher, Jean-Philippe. «Un système adaptatif par agents avec utilisation des algoritmes génétiques : application à l'ordonnancement d'atelier de type job-shop nm.» Thèse de doctorat, Le Havre, France, 2000.

Wang, Xiaoqing, Hongwei Ding, Minmin Qiu, et Jin Dong. «A low-carbone production scheduling systemconsidering renewable energy.» *Service Operations, Logistics, and Informatics (SOLI), 2011 IEEE International Conference on.* Beijing, China: IEEE, 2011. 101-106.

Watson, Jean-Paul, J.Christopher Beck, Adele E Howe, et L.Darrell Whitley. «Problem difficulty for tabu search in job-shop scheduling.» *Artificial Intelligence* 143, n° 2 (2003): 189-217.

Xing, Li-Ning, Ying-Wu Chen, et Ke-Wei Yang. «Multi-objective flexibl job shop schedule: design and evaluation by simulation modeling.» *Applied Soft Computing* 9, n° 1 (2009): 362-376.

Yamada, Takeshi, et Ryohei Nakano. «A Genetic Algorithm Applicable to Large-Scale Job-Shop Problems.» *Parallel Problem Solving from Nature 2, PPSN-II.* Editeur R.Männer and B. Manderick, 1992. 281-290.

Yildirim, Mehmet Bayram, et Gilles Mouzon. «Single-Machine Sustainable Production Planning to Minimize Total Energy Consumption and Total Completion Time Using a Multiple Objective Genetic Algorithm.» *IEEE Transactions on Engineering Management* 59, n° 4 (2012): 585-597.

Yiyong, He, Weng Wei, et Fujimura Shigeru. «Improvements to genetic algorithm for flexible job shop scheduling with overlapping in operations.» *16th International Conference on Computer and Information Science (ICIS).* Wuhan, China: IEEE, 2017. 791-796.

Zeng, Chengkuan, Jiafu Tang, et Chongjun Yanb. «Scheduling of no buffer job shop cells with blocking constraints and automated guided vehicles.» *Applied Soft Computing* 24 (2014): 1033-1046.

Zhang, Liping, Xinyu Li, Liang Gao, Guohui Zhang, et Xiaoyu Wen. «Dynamic scheduling model in FMS by considering energy consumption and schedule efficiency.» *Computer Supported Cooperative Work in Design (CSCWD), 2012 IEEE 16th International Conference on.* Wuhan, China: IEEE, 2012.

Ziaee, Mohsen. «A heuristic algorithm for solving flexible job shop scheduling problem.» *The International Journal of Advanced Manufacturing Technology* 71, n° 1 (2014): 519–528.

Zitzler, Eckart, et Lothar Thiele. «An evolutionary algorithm for multiobjective optimization: the Strength Pareto Approach.» *IEEE Trans. on Evolutionary Computation.* 2000. 257-271.

Zitzler, Eckart, Marco Laumanns, et Lothar Thiele. «SPEA2: Improving the Strength Pareto Evolutionary Algorithm for Multiobjective Optimization.» *Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems. Proceedings of the EUROGEN'2001.* Athens. Greece, 2001.

Zribi, N, A El-Kamel, et P Borne. «Minimizing the makespan for the MPM job-shop with availability constraints.» Édité par Elsevier. *International Journal of Production Economics* 112, n° 1 (2008): 151-160.

Zribi, Nozha, Imed Kacem, Abdelkader El Kamel, et Pierre Borne. «Assignment and Scheduling in Flexible Job-Shops by Hierarchical Optimization.» Édité par IEEE. *IEEE Transactions on Systems, Man, and Cybernetics* 37, n° 4 (2007): 652-661.

Table of Contents

International Journal of Information Systems and Supply Chain Management

Volume 11 • Issue 4 • October-December-2018 • ISSN: 1935-5726 • eISSN: 1935-5734
An official publication of the Information Resources Management Association

Research Articles

The *International Journal of Information Systems and Supply Chain Management* is indexed or listed in the following: ACM Digital Library; Australian Business Deans Council (ABDC); Bacon's Media Directory; Cabell's Directories; Compendex (Elsevier Engineering Index); CSA Illumina; DBLP; GetCited; Google Scholar; IAOR Online; INSPEC; JournalTOCs; Library & Information Science Abstracts (LISA); MediaFinder; Norwegian Social Science Data Services (NSD); SCOPUS; The Index of Information Systems Journals; The Standard Periodical Directory; Ulrich's Periodicals Directory; Web of Science; Web of Science Emerging Sources Citation Index (ESCI)

International Journal of Information Systems and Supply Chain Management
Volume 11 • Issue 4 • October-December 2018

Predictive Reactive Approach for Energy-Aware Scheduling and Control of Flexible Manufacturing Processes:
Application on Flexible Job Shop

Mohammed El Amine Meziane, Université Oran1, Ahmed Ben Bella, Oran, Algeria
Noria Taghezout, Université Oran1, Ahmed Ben Bella, Oran, Algeria

ABSTRACT

Manufacturing processes are responsible for a considerable amount of global energy consumption and world CO_2 emissions. Reducing energy consumption during manufacturing is considered one of the most important strategies in contributing to the green supply chain. In this context, the authors propose a new predictive-reactive approach to control energy consumption during manufacturing processes. In addition to forecasting the energy needs, the proposed approach controls the uncertainty of energy volatility and limits energy waste during manufacturing processes. With the integration of this economic-environmental manufacturing efficiency in supply chains, and controlling uncertainty, this approach positively contributes to green and agile supply chains. A multi-objective genetic algorithm (NSGA-2) is proposed as a predictive method, and a new reactive method is developed to dynamically control the energy consumption throughout the peak energy consumption in real time. The approach was tested on the AIP-PRIMECA benchmark, which reflects a real production cell.

KEYWORDS

Energy-Aware Manufacturing Scheduling and Control, Green and Agile Supply Chain, Multi-Objective Optimization, Non-Dominant Sorting Genetic Algorithm-2, Predictive-Reactive Approach

INTRODUCTION

As stated by the United Nations, 40% growth in the world population is predicted by 2050. As a result, energy consumption (electricity and other resources) will double in the next 40 years and electricity use only, is forecasted to double in 18 years. In addition, to avoid serious climatic changes, climate specialists from the International Energy Agency[1] estimated that it is crucial to reduce by half the carbon dioxide emissions.

Since 1980, saving energy has attracted the attention of a very large number of researchers. All these studies found that energy efficiency is the widest and cheapest resource. However, interesting opportunities can be explored for energy efficiency. These opportunities reside in the supply chain for the industry, where energy is used to manufacture the parts and suppliers that go into the production of good. This applies to every single good in the market, whether those sold to consumers or production facilities. In fact, a supply chain management (SCM) "…is a network of facilities and distribution options that perform the functions of procurement of materials, transformation of these materials into

DOI: 10.4018/IJISSCM.2018100103

the intermediate and finished products and the distribution of these finished products to customers…" (Rajesh, Pugazhendhi, & Ganesh, 2011).

To illustrate our contribution for supply chain energy management, we focus on the auto parts manufacturing, which provides great energy savings, where suppliers sell parts to car manufacturers, assembling them into a final product. One of the well-known industrial models for parts manufacturing is the Job shop model.

The considered Job shops are small manufacturing systems that handle job production, that is, custom/bespoke or semi-custom/bespoke manufacturing processes. Machines are aggregated in shops by the nature of skills and technological processes involved, each shop may contain different machines. Jobs aren't necessarily constrained to a single machine which gives this production system processing flexibility. Machine-shop is a typical example of job shop, which may make parts for local industrial, farm machinery and implement, beats and ships, or even batches of specialized components for the aircraft industry. Grinding, honing, jig-boring, gear manufacturing are other types of common job shops. Job shops have many advantages: high expansion flexibility (machines are easily added or substituted, high flexibility in production engineering and high robustness to machine failure. However, scheduling of job shops is very hard due to high product variability and twisted production flow.

One of the key sectors for major industrialized countries in the world is the global auto industry, thus, the complexity of the industry must be well-managed to generate profit. During the last 30 years, the production of vehicle components has been outsourced to suppliers (The processes of car manufacturing are distributed across a supply chain spanning multiple plants, companies, and even countries). Sometimes, hundreds of companies are involved in the supply components of a given vehicle, from pistons to windshield wipers. The overall business efficiency of building a car is impacted by each single supplier. The global automotive component market generates more than 1$ trillion in 2017, according to Global Industry Analysts[2]. This means that, to remain globally competitive, automotive manufacturers will have to reduce the energy consumed by their process, essentially, not just at their manufacturing facilities, but also at all their suppliers' facilities.

While the implementation of the predictive approach for part manufacturing processes reduces their energy consumption, the cumulative energy consumption of car manufacturer's suppliers will be greatly reduced, which contributes limiting the overall cost of vehicle production. According to U.S. Department of Energy's Argonne National Laboratory[3], 10% energy saving per vehicle at the average production rate of 5 million cars per year, allows saving $236 million annually (Boville & Dussault, 2012).

The reliability of the energy supply drops because of the complex and dynamic environment of supply chain, such as the increase of both demand for energy and prices (Katiraee, Shirazi, & Fazlollahtabar, 2017). For example, the U.S.A is the largest single electricity-consuming country, using a quarter of the world's electricity and the cost of just one hour of downtime reaching up to $1.2 million. The proposed reactive approach can control in real-time energy consumption during parts manufacturing processes, by reducing consumption when energy supply drops, in order to avoid downtime of any car manufacturer's supplier. This approach not only controls, energy consumption in real-time, but it provides energy use data to discover where in the manufacturing process energy is being used and make the changes needed to reduce it. The energy-aware sensor-based model increases intelligence at the manufacturing processes level for reducing energy wastage by turning off machines when they are not working. In the second following section, we present a state of art on energy-aware manufacturing scheduling and control.

STATE OF ART: ENERGY-AWARE MANUFACTURING SCHEDULING AND CONTROL

Recently, studies on energy-aware manufacturing scheduling have been significantly increased. According to (Prabhu, Trentesaux, & Taisch, 2015), energy-aware manufacturing operations are an

emerging and highly challenging concept. They defined this concept as "…a manufacturing operations management system that considers in a predictive or reactive way, and in addition to usual production decision variables, objectives and constraints (time based or quality based), the energy as a decision variable, an objective or as a constraint."

Three main challenges can be found when taking into account energy-aware manufacturing operations (Prabhu, Trentesaux, & Taisch, 2015):

1. Energy-efficiency vs. manufacturing systems effectiveness;
2. Volatility in energy availability, supply and cost;
3. Modelling energy consumption.

We consider in this paper the two first challenges.

In the first challenge (energy-efficiency vs. manufacturing systems effectiveness), two performance indicators should be considered. However, considering that energy in manufacturing operations management increases significantly the complexity of classical problems (e.g. job shop scheduling problem is already classified as NP-hard). (Roghania, Rasli, Gheysar, & Gheysar, 2012) defined the energy-efficiency as seeking for the best use of meanwhile effectiveness means seeking for the best results. However, efficiency and effectiveness are generally conflicting objectives; for example, the performance of time operations scheduling may be degraded when reducing the energy consumption. There exist three strategies to face this challenge (Patch, Berger, Sallez, Bonte, & Adam, 2014):

1. Energy-efficiency can be optimized using manufacturing-operations effectiveness as constraint (e.g. opportunistic energy saving: machine on/off);
2. Optimizing manufacturing-operations effectiveness under energy constraint (e.g. optimize effectiveness under the total available energy constraint like the peak load);
3. Seeking for a compromise between energy-efficiency and manufacturing-operations effectiveness.

Volatility of energy availability, supply and cost of energy are the main factors of the second challenge of energy-aware manufacturing systems. These factors require more reactive management systems (Pouya, Sami, & Bernard, 2015). The emergence of new sources of energy (e.g. photovoltaic panels, wind turbines) made the cost and the availability of energy more unpredictable. Consequently, the energy pricing becomes more dynamic. All these factors were considered recently as sources of uncertainty in manufacturing environment. (Christian, Cyril, & Marie-Claude, 2002) classified uncertainty into three categories according to their knowledge level: completely unknown uncertainty (e.g. accident in the place of work), doubts of the future (we take into account the volatility of energy supply) and partially known uncertainties (machine breakdown). This impels us to analyse the different approaches of manufacturing optimization scheduling to face the both cited challenges.

An interesting classification of scheduling under uncertainty was proposed by (Tarek, Sondes, Nassima, & Damien, 2014). Generally, researchers adopt either predictive or reactive approaches to resolve the problem of energy efficiency in manufacturing operations. The adopted approaches can be qualified as exact or approximate.

The first widely proposed approaches were exact and predictive (off-line scheduling), which focused on optimization of power consumption, using typically mathematical models. (Fang, Uhan, & Sutherland, 2013) optimized the total processing time of operations with power consumption constraint for energy efficiency. Two formulations based on the exact method MIP (Mixed Integer Programming) were developed. Many other exact methods were proposed: fractional mixed integer linear programming (Wang, Ding, Qiu, & Dong, 2011), integer linear programming (Zhang, Li, Zhang, & Wen, 2012) and mixed integer linear programming (Bruzzone, Anghinolfi, Paolluci, &

Ronelli, 2012). The peak power consumption was the first aspect of energy that was tackled. The developed approach by (Bruzzone, Anghinolfi, Paolluci, & Ronelli, 2012) was based on two phases. The energy saving was considered in the second phase and the initial schedule was refined by mixed integer linear programming to control peak power consumption. The footprint was taken into account in addition to the power consumption in (Fang, Uhan, & Sutherland, 2011).

However, due to the highly time-consuming nature of these mathematical models, a second generation appeared which is based on meth heuristics (e.g. taboo search, genetic algorithm). These approaches provide solutions of good quality in reasonable time. Combination of exact and approximate approaches was proposed by several authors. (Yildirim & Mouzon, 2012) proposed a technique to save energy. They stopped machines when they aren't working. A mathematical model was developed to control the energy consumption and the total workload of one machine. A multi-objective genetic algorithm was used to get a set of non-dominated solutions; however, this approach was limited to one machine.

Later, (Luo, Du, & Huang, 2013) developed an ant colony algorithm for the flow shop scheduling problem. The considered objective functions are the production time and the cost of power with prices of different utilization periods. Firstly, scheduling is calculated by an ant colony algorithm then, a procedure based on right shifting of operations was used to adjust the power cost.

The authors (Dai, Tang, Giret, Salido, & Li, 2013) created an improved genetic-simulated annealing algorithm, which economizes energy and reduces production time (makespan). The applied tests on metal manufacturing showed the capacity of this approach to identify optimal Pareto front and revealed that the relationship between makespan and energy is conflicting.

A multi-objective optimization model was proposed by (Jiang, Zuo, & Mingcheng, 2014) for flexible job shop problem. Four objective functions were taken into consideration: energy consumption, total production time, the processing cost and the weighted cost of quality. They used NSGA-2 algorithm in which, Blood-variation was proposed to avoid a premature convergence.

Avoiding the periods in which, the cost of energy is expensive, the authors (Shrouf, Ordieres-Meré, & García-Sánchez, 2014) reduced the cost of energy. The decisions are made at the level of machines: the starting processing time of jobs, when machines will be set to standby or ready state. The proposed model generates scheduling with the cheapest production plan. A genetic algorithm was used to obtain near optimal solutions. The results indicate that significant reductions in energy costs can be achieved by avoiding high-energy price periods.

The pervious Meta heuristics approaches proposed in this second generation are faster than the exact ones. In fact, the mathematical models don't match the reactivity required in a dynamic and uncertain context (Marik & McFarlane, 2005). However, authors in the second generation don't pay too much attention to the dynamic context of energy-aware manufacturing processes. For example, they focus only on static peak power and don't deal with real-time power consumption control. As suggested by (Trentesaux & Prabhu, 2014) for scheduling in highly dynamic context, reactive approaches are promising for power consumption and overall power limitation. (Prabhu & Taisch, 2012) controlled the power consumption by the heuristic: "distributed arrival time control" for one machine. (Patch, Berger, Sallez, Bonte, & Adam, 2014) proposed a potential fields model to reduce reactively the overall power consumption of manufacturing system. However, this study was limited to the energy wastage and didn't pay attention to the control of overall power consumption according to power supply. Later, (Pach, Berger, Sallez, & Trentesaux, 2015) proposed a reactive method based on potential fields, as an extension of the previous work for dynamic operations scheduling. This technique considers both efficiency and effectiveness. The assignment and routing decisions are made at the level of products, depending on the attractive potential fields emitted by machines. The potential fields emitted by products determine the state of machines.

According to the previous works related to energy-aware manufacturing scheduling and control, the adopted approaches are either predictive or reactive. Predictive approaches generate a deterministic scheduling (off-line). Nevertheless, when the manufacturing environment is dynamic; the off-line

scheduling becomes rapidly obsolete in the new state of manufacturing system. In this case, the reactive approaches are suitable to control reactively the manufacturing operations. But, the reactive approaches privilege the speed of decisions on their overall quality. In this context, adopting a hybrid approach (predictive-reactive) allows us to take the advantage of both static and dynamic behavior.

Predictive-reactive approach is considered as one of the approaches supporting risks, like energy supply disturbance. This approach requires two phases: in the first phase, a deterministic off-line scheduling is calculated then, this schedule is adapted on-line in the second reactive phase to absorb disturbance like degradation of the available energy.

In this paper, a predictive-reactive approach is suggested to control the overall energy consumption, taking into account both effectiveness (makespan) and efficiency (energy consumption and peak of energy). This approach was tested on AIP-PRIMECA cell (Trentesaux, et al., 2013) which reflects a real production cell modeled as flexible job shop. The obtained results are compared with the reactive approach proposed by (Pach, Berger, Sallez, & Trentesaux, 2015).

Another crucial aspect in energy-aware manufacturing is reducing energy wastage. According to (Giret, Trentesaux, & Prabhu, 2015), several companies suffer from their manufacturing systems which become oversized compared to market needs. Turning off machines when they aren't processing any operation, allows saving a considerable energy. In this context, we proposed an energy-aware sensor-based model for energy wastage control. Before presenting our contribution of the predictive-reactive model to the agile and green supply chains (Figure 1), a mathematical formulation of the considered problem is presented in the following section.

FORMULATION OF THE PROBLEM

In this section, we present the mathematical model of flexible job shop scheduling problem (FJSSP) with the constraints for feasible schedules and transport time between machines. In FJSSP, we try to organize the execution of N jobs $j \in \left\{J_1,\ldots,J_N\right\}$ on M machines $k \in \left\{M_1,\ldots,M_m\right\}$. Each job J_j is composed of a set of n_j ordered operations $\left\{O_{j,n_1},\ldots,O_{j,n_j}\right\}$ (precedence constraint). Each operation must be executed by one machine. The execution of the operation $O_{j,i,k}$ (operation number i of the job j) on the machine M_k makes this machine unavailable for the other operations for a certain period $P_{j,i,k}$ (resources constraint). If any operation can be executed by any machine, the flexibility is total. However, if the candidate machines of an operation represent a subset of the total number of machines, the flexibility is partial. In addition to the classical constraints, we take into account the transport time between machines. A job J_j is transported via shuttle (automated guided vehicle) from a machine M_{m1} to a machine M_{m2} with a transport time $T_{j,\,m1-\,m2}$. We aim to minimize the transport time between machines.

The classical effectiveness indicator is the makespan (Cmax) which reflects the total completion time of jobs. Jobs are transported by shuttles from a machine to machine until they will be achieved. Thus, the makespan is calculated by Equation (1):

$$Cmax = \max_{1\le j\le N} tfa_j$$

$$t_{j,i,k'} = max\left(tf_{j,i-1,k} + Tr_{kk'},\ tf^{-1}{}_{j,i,k'}\right) \quad (1)$$

$$tf_{j,i,k'} = t_{j,i,k'} + P_{j,i,k'}$$

- tfa_j : The finishing time of the last operation in the job *j*.

$tf^{-1}_{j,i,k'}$: The finishing time of the precedent operation of $O_{j,i,k'}$ in the machine $M_{k'}$.
$P_{j,i,k'}$: The processing time of the operation *i* of the job *j* by the machine $M_{k'}$.
$tO_{j,i,k'}$ & $tfO_{j,i,k'}$: The starting and the finishing times of $O_{j,i}$ performed by the machine $M_{k'}$.
$O^{-1}_{j,i,k'}$ & $O^{+1}_{j,i,k'}$: The precedent and the next operation of $O_{j,i}$ in the queue of the machine $M_{k'}$.
$O_{j,i-1,k'}$ & $O_{j,i+1,k'}$: The precedent and the next operation of $O_{j,i}$. The three operations belong to the same job j. $Tr_{k'k}$ represents the transport time between the machines *k'* and *k*.

This classical problem is classified as NP-hard and there is no exact method which guarantees optimal solutions. Energy efficiency is considered in this paper. Thus, the efficiency performance indicator is added to the effectiveness. The resolution of this problem becomes harder than the classical one.

THE PROPOSED PREDICTIVE-REACTIVE APPROACH FOR THE CONTROL OF ENERGY

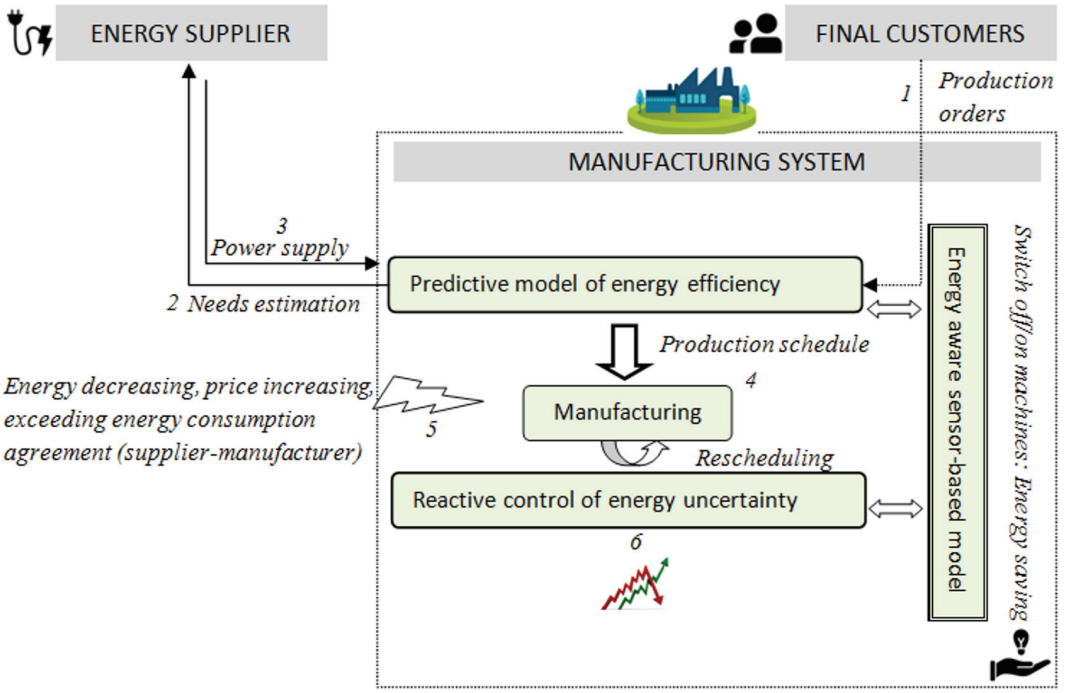
Phase 1: A Predictive Method Based on NSGA-2 for Energy Efficiency

We propose a predictive method for energy efficiency using the multi-objective genetic algorithm NSGA-2 (Non Dominant Sorting Genetic algorithm). Three criteria are considered for the evaluation of chromosomes (coded solutions): Cmax (makespan), the overall energy consumption and the peak of energy consumption. Once the set of Pareto optimal solutions is achieved, the decision maker determines the weight of each objective function using the technique AHP: Analytic Hierarchy Process (Saaty & Thomas, 2008) according to the actual manufacturing situation. For example, if the decision maker receives urgent production orders then, the coefficient of makespan should be increased. However, when a possible volatility of the available energy supply should be predicted, the decision maker can increase the coefficient of both the overall energy consumption and the peak of energy consumption. The selection of one solution among the set of Pareto optimal solutions is determined by Equation (2):

$$Obj_i = W_C \frac{C_{max} - C_i}{C_{min} - C_{min}} + W_E \frac{E_{max} - E_i}{E_{max} - E_{min}} + W_P \frac{P_{max} - P_i}{P_{max} - P_{min}} \quad (2)$$

W_C, W_E, W_P are the coefficients of Cmax, energy consumption and the peak of energy consumption. C_i, E_i, P_i are the values of Cmax, energy consumption and the peak of energy consumption of the solution i^{th} among a set of Pareto optimal solutions.

Figure 1. Contribution of the predictive-reactive model to the agile and green supply chains



The Model of Energy Efficiency

This model is based on the three cited functions. Thus, the optimization of scheduling can be represented as follow: Min (*C, E, P*). *C* represents the Cmax, *E* the energy consumption and *P* the peak of energy consumption. The three objective functions are calculated as follow:

1. *Cmax*: the total time to achieve all jobs (Equation (1));
2. The overall energy consumption.

It is the consumed electric power (watts) by each machine during the processing. The overall energy consumption is presented in Equation (3):

$$E = \sum_{k=1}^{M} \sum_{j=1}^{n} \sum_{i=1}^{a_i} e_k * P_{j,i,k} \quad (3)$$

e_k is the energy consumption per second, by the machine *Mk*. $P_{j,i,k}$ is the processing time of the operation *O j,i* by the machine *Mk*.

3. The peak of energy consumption.

It represents the maximum energy consumption per second Equation (4):

$$P = \max_{t=1,..C\max} E_t \tag{4}$$

$$E_t = \sum_{k=1}^{M} E'_{k,t}$$

$E'_{k,t}$ is energy consumption of the machine k at the moment t.

The Scheduling Algorithm: NSGA-2

Many approaches for multi-objective optimization were proposed in the literature. Usually, the used methods are: single objective transformation, random weighting and Pareto method (Xing, Chen, & Yang, 2009). The Pareto method is preferred by researchers because it allows obtaining a set of Pareto optimal solutions in optimizing process, which is coherent with an actual scheduling problem (Ghasem & Mehdi, 2011). Many representative algorithms exist, such as MOGA (Fonseca & Fleming, 1993), NSGA and NSGA-2 (Srinivas & Deb, 1995) (Deb, Patrap, Agarwal, & Meyarivan, 2002), PESA and PESA-2 (Corne, 2000). However, because of its better distribution and fast convergence speed, NSGA-2 is widely used.

Chromosome Coding

Each solution is coded as pairs (machine - operation): {(M2, O3,1), (M1, O1,1), (M3, O3,2), (M3, O3,2), (M1, O3,2), (M2, O1,2)}.

This coding reflects assignment and routing of operations. It is simple and fast to decode it.

The Selection Operator (Elitism Non-Dominated Sorting Algorithm)

Selection nature of multi-objective genetic algorithm is more complex than the mono-objective one. This operator divides individuals into ranks and applies a strategy to evaluate and sort the individuals of the same rank:

- Ranking of Individuals:
  - There are two types of Pareto sorting methods which are widely used: the recursion sorting (Jensen, 2003) method and the adopted quick sorting method (Chen, Song, Zheng, Zhao, & Yao, 2010). According to recent studies, the second method has better computational performance.
- Selection strategy of non-dominated solution with the same rank:
  - The application of non-dominated sorting generates individuals with different ranks, and those with low rank will be chosen to participate in the evolution. For the individuals of the same rank, some strategies ensure diversity of the colony during evolution. Recently, many strategies have been proposed, which concerns information entropy, density-based clustering, niche technology and classification (Jiang, Zuo, & Mingcheng, 2014). The adopted strategy is the density based on clustering. This strategy can capture both of diversity and distribution of population macroscopically and make the relationship among individuals with capability of controlling the colony during the evolution process (Chen, Song, Zheng, Zhao, & Yao, 2010). The algorithm of Crowding-distance (I) is used to sort solutions of the same rank.

The solution i is better than the solution j if:

1. $i_{rank} < j_{rank}$; or
2. $i_{rank} == j_{rank}$ and $i_{distance} > j_{distance}$.

Algorithm 1. Crowding-distance (*I*) (Deb, Patrap, Agarwal, & Meyarivan, 2002)

Begin
L = $\left|I\right|$ // the number of solutions in the set *I*
For each i, set $I\left[i\right]_{distance} = 0$ // initialize the distance of each solution
For each objective m
$I = sort\left(I,m\right)$ // sort solutions according to the value of each objective function
$I\left[1\right]_{distance} = I\left[L\right]_{distance} = \infty$ // the boundaries solutions are always taken
For i = 2 to (L-1) // for all the other solutions
$I\left[i\right]_{distance} = I\left[i\right]_{distance} + \left(\left(I\left[i+1\right]*m\right) - \left(I\left[i-1\right]*m\right)\right) / \left(f_m^{max} - f_m^{min}\right)$
End-for
End-for
End-for
End

Note: $I\left[i\right]*m$ is the value of the m^{th} objective function of the solution *i* in the set of solution *I*.
f_m^{max} , f_m^{min} : the maximum and the minimum values of the m^{th} objective function.

If two solutions belong to the same rank, the solution belonging to the less crowded zone is the best one. With regard to the diversity of the population, individuals with a larger crowding distance have a big probability to participate to reproduction and evolution.

Crossover Strategy

This operator exchanges information from each pair of chromosomes (parents) to get two new individuals (better descendants). Respecting the precedence constraints on each job is necessary. However, avoiding a correction process on the generated solutions, which increases dramatically the calculation time, will be more interesting. This is why we have chosen the crossover operator developed by (Lee & Yamaha, 1998), which respects the problem's constraints.

Mutation Strategy

- Assignment Mutation:
  - First, we choose two different indexes L1 and L2 from the interval $\left\{1,\rho\right\}$ such as ρ is the total number of operations. Then, we choose randomly for each gene between L1/L2 another machine from the set of the alternative machines.
- Sequencing Mutation:
  - We choose randomly two different operations (from different jobs) then, we permute these two operations.

Phase 2: The Reactive Algorithm for Energy Control

Recently, the reactivity of manufacturing systems becomes crucial strategy: the volatility and unpredictability of energy availability as well as the instability of supply and energy costs (Ghadimi, Kara, & Kornfeld, 2015). For example, carbon emissions are important during periods of peak electricity due to the use of more expensive and less clean resources (Prabhu V., 2012).

In the same context, energy supply agreements between the customer and the supplier must be respected and any additional consumption penalizes the manufacturer. Moreover, the electricity pricing can be dynamic.

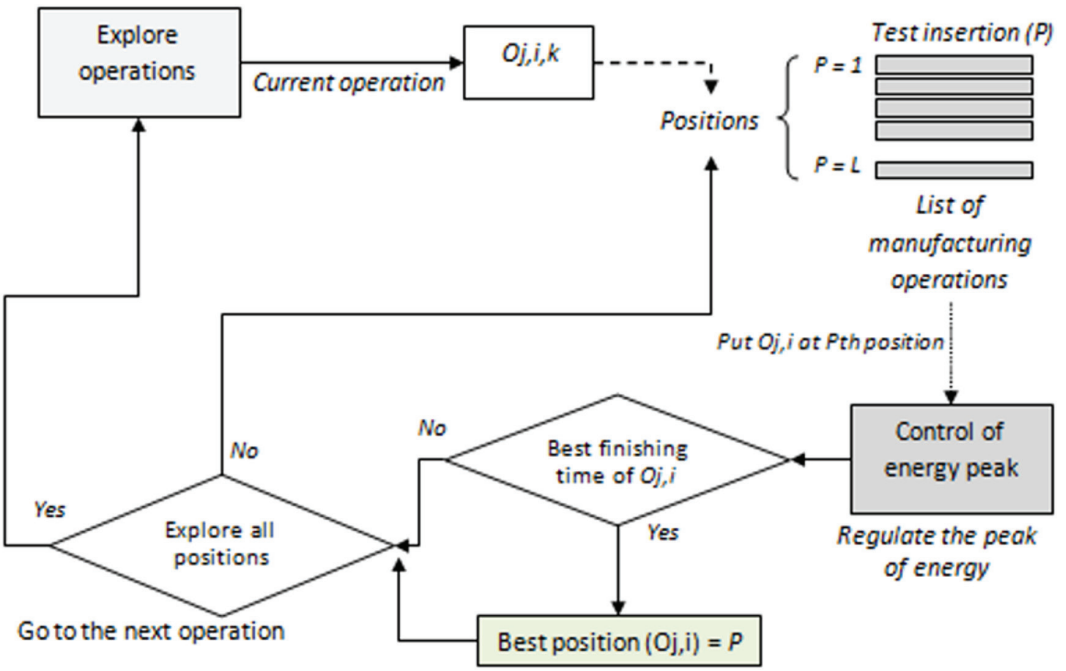
Among other recently emerging factors that accentuated the unpredictability of energy cost consumption as well as its availability and volatility of its supply is the increasing use of solar panels and wind turbines as new sources of energy. The evolution of energy availability should be predicted but, the price and consumption behavior can be difficult to predict (Borlase & Fan, 2009). Consequently, the definition of exact strategies for manufacturing systems in hardly predictable environment is a hard task. Many methods exist to resolve this challenge such as, analyze data tools, risk management tools, simulation tools or reactive algorithms (e.g. fast heuristics, multi-agent architectures) (Prabhu, Trentesaux, & Taisch, 2015).

We propose in this paper a reactive algorithm (Figure 2) to control dynamically the peak of energy consumption and optimize the production time during manufacturing process. This algorithm is applied during the execution of the scheduled operations, generated by the genetic algorithm NSGA-2 (predictive method).

The Figure 2 shows the proposed model which controls dynamically the peak of energy consumption and optimizes the production time. The dynamic control of energy peak is required in many cases:

- Decreasing energy supply: the use of new energy technologies such as photovoltaic panels, wind turbines. These new sources of energy are environmentally friendly but not stable;
- Instability in energy pricing: in some cases, the energy provider may increase the price of energy, especially during the periods of peak;

Figure 2. The proposed model for energy consumption peak with dynamic control



- Provider-customer energy consumption agreement: a customer must regulate the energy consumption during manufacturing process to avoid any penalty. These events can occur at any moment during manufacturing process, which requires a fast reactive strategy to regulate dynamically the peak of energy consumption.

Whenever new energy threshold is required, the model reschedules manufacturing operations by the following steps:

1. Insert each operation into a position and regulate the peak of energy consumption using the algorithm: control_pic_energy(*S, peak, t1, t2*) which regulates dynamically the peak of energy consumption;
2. Choose the best position that corresponds to the earliest completion time of the current operation. Repeat the first two steps until all operations are completed. This procedure seeks the best routing of each operation by inserting it at a position. Afterwards, the next loop recalculates the starting and the finishing times of the operations to regulate the peak of energy consumption (apply a right shifting on *oj,i* until the new threshold is achieved). The procedure is repeated for all positions and the one that allows the earliest completion time will be retained. See Algorithm 2.

The peak_interval (*S, t1, t2*) algorithm (Algorithm 3) measures the peak of energy consumption in a time interval delimited by [*t1*, *t2*]. This algorithm works with a function called energy_consumption_per_second that measures the peak of energy at a given moment.

The energy_consumption_per_second algorithm (Algorithm 4) returns the energy peak at a given moment. A machine in function mode consumes 400 watts and 250 watts when it isn't working.

After presenting the multi-objective optimization model for dynamic control of energy consumption, we propose in the following section an energy-aware sensor-based model to optimize energy wastage, when the machines are waiting for shuttles transporting products to be assembled.

Algorithm 2. control_pic_energy(*S, peak, t1, t2*)

Begin
For each $O_{j,i}$ // explore each operation
For P = 1 to L // the different positions of operations, L is the last position
Insert ($O_{j,i}$; P) // insert the operation $O_{j,i}$ in P^{th} position
$tO_{j,i,k} = Max\left(\left(tfO_{j,i-1} + transport\_time\right), tfO^{-1}_{j,i}\right)$ // starting time
$tfO_{j,i,k} = tO_{j,i,k} + P_{j,i,k}$ // finishing time
Shift = 0 // initialize the operations right shifting parameter.
//check if Oj,i belongs to t1,t2 and if the new peak is exceeded
while $Min\left(t_2 , tfO_{j,i,k}\right) - Max\left(tO_{j,i,k} , t_1\right) > 0 \;\&\&\; peak\_interval\left(S , t_1 , tfO_{j,i,k}\right) > peak$
$tO_{j,i,k} = Max\left(\left(tfO_{j,i-1} + transport\_time\right), tfO^{-1}_{j,i}\right) + \text{Shift}$ // new starting time
$tfO_{j,i,k} = tO_{j,i,k} + P_{j,i,k}$ // the new starting time of $O_{j,i}$ is shifted by one second.
Shift = shift + 1; // increment the shifting by one second.
End.

Algorithm 3. peak_interval(*S, t1, t2*)

```
Begin
current_peak = max_peak = 0
for (i = t1 to t2) // explore the interval of time between the second t1 and t2
current_peak = energy_consumption_per_second (S, i, i+1)
if (current_peak > max_peak)
max_peak = current_peak
End.
```

Algorithm 4. energy_consumption_per_second (*S, i, i+1*)

```
Begin
Energy_consumption = 0
For each machine k'
For each operation Oj,i,k
If (k'= k)
If( tO_{j,i} ≤ t_1 && tfO_{j,i} ≥ t_2 )
Energy_consumption = Energy_consumption + 400 // 400 watts is the energy consumption in working state.
If (Energy_consumption = 0)
Energy_consumption = Energy_consumption +250 //250 watts is the consumed energy by a machine when it isn't working.
End.
```

ENERGY WASTAGE CONTROL

This model is based on the case study AIP-PRIMICA Cell (Trentesaux, et al., 2013). This production cell is composed of five machines:

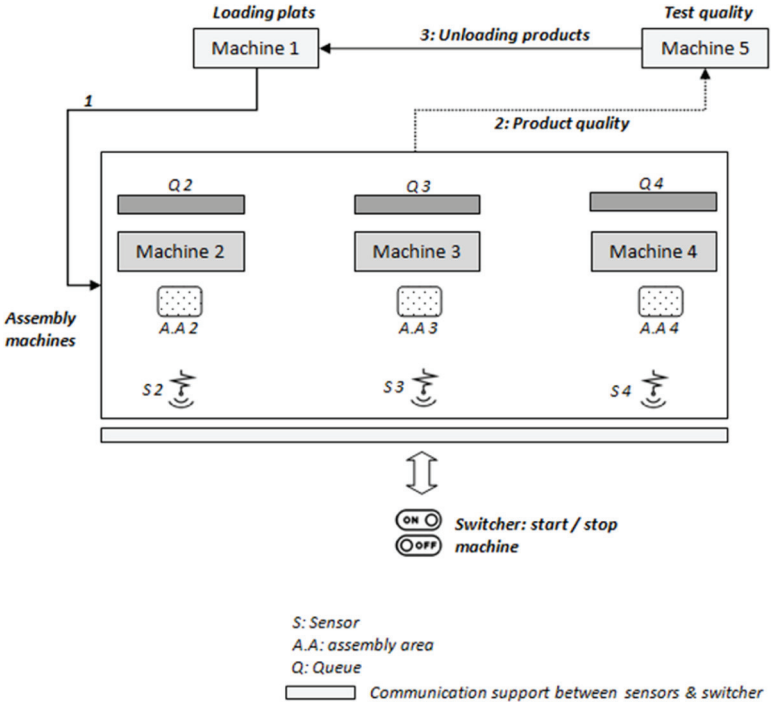
Machine 1: Loading of empty dishes and unloading of dishes filled with finished products.
Machine 2, 3 and 4: Machines for assembling parts in order to obtain finished products.
Machine 5: When the machines 2, 3 and 4 complete the assembly, the machine 5 performs a quality test of the finished products, if an abnormality is detected by this machine, a manual maintenance operation will be initiated.

The maintenance is the only manual operation in the production cell. Maintenance is not considered in this study. Most of the operations are carried out by the assembly machines (machine 2, 3 and 4). These machines have three elements:

1. **A waiting list (queue):** Containing the unfinished products waiting for assembly;
2. **Assembly area:** Each product is transported by an automatic conveyor. When the waiting list for an assembly machine is empty, conveyor is positioned in the assembly area;
3. **Sensor:** The role of sensor is to capture any conveyor in the queue and the assembly area. A communication support between the sensor of each assembly machine and the switcher ensures the transmission of information (Figure 3).

The switcher receives the transmitted information from the sensor of each assembly machine and reacts. If the queue list of an assembly machine and its assembly area are free, the switcher stops this assembly machine. Once a product is on its queue list, the switcher starts the machine. In this way, energy can be saved. Thus, energy control can be achieved by scheduling operations and

Figure 3. The proposed energy-aware sensor-based model for energy wastage control



controlling physically the machines. The combination of the two models is possible, but for reasons of simplification, the two approaches have been separated.

CASE STUDY: AIP PRIMECA FMS

Benchmarks are used to compare the systems performance. Many benchmarks are proposed in the operation research literature. (Beasley, 1990), (Taillard, 1993) and precisely, (Brandimarte, 1993) for flexible job shop scheduling problem. However, these benchmarks are theoretical and don't satisfy the constraints of the real manufacturing systems like transport time and energy consumption.

We choose a special Benchmark, based on a real case study: AIP-PRIMECA CELL, which was developed by the University of Valenciennes, France (Trentesaux, et al., 2013)[4]. In addition, it takes into consideration the transportation times and energy consumption which are very important in the real industrial environment and generally ignored (Belkaid, Yalaoui, & Sari, 2016). Three products are proposed: ''BELT'', ''AIP''' and ''LATE''. Thus, a product is a subset of jobs, among the seven possible job types, corresponding to different arrangement of letters. We propose simulations using four kinds of products orders: ''1 x AIP'', ''2 x AIP'', ''3 x AIP'' and ''BELTAIP''.

EXPERIMENTS AND RESULTS

In order to evaluate our proposed approach for energy control, a series of experiments has been conducted. In the first experiment (Table 1), an initial solution was generated randomly. Then, this solution was compared with the one, obtained by the genetic algorithm (NSGA-2).

In this first experiment (Table 1), two objective functions were considered: Cmax (production time in seconds) and total energy consumption (watts). The coefficients of each function are set

Table 1. Energy efficiency results using the multi-objective genetic algorithm and energy wastage control algorithm

	Initial Solution		NSGA-2		Gain		Energy Wastage Control	Gain
Production Order	Cmax (s)	Energy (wh)	Cmax (s)	Energy (wh)	Cmax %	Energy %	Energy (wh)	Energy %
1xAIP	486	68	239	37.33	49.17	20.13	33.3	51.02
2xAIP	707	98.98	355	71.11	49.78	11.94	66.2	33.11
3xAIP	873	136.45	449	104	48.56	6.68	100	26.80
6xAIP	1592	234.37	829	204	47.92	4.05	200	14.66
1xBELT	512	93.54	348	57.77	32.00	7.80	53.33	43.00
1xBELT 1xAIP	860	121.25	409	91.11	45.10	9.26	86.66	28.52
2xAIP 1xLATE	929	162.70	509	124	45.20	7.45	120	26.24

to 0.5. The solutions of different production orders obtained by NSGA-2 are much better than the initial solutions. The last two columns present energy consumption and energy gain, using the energy wastage control model.

Next, the reactivity of the proposed algorithm for controlling energy consumption is studied. This experiment is for the production order 2xAIP and it takes into account the scenario in which, at any given moment a threshold of energy consumption is imposed. As mentioned in section 4, the causes of this scenario are multiple. Figure 4 shows the Gantt diagram and energy consumption chart without energy threshold for the manufacturing order 2xAIP.

Initially, the three machines are waiting for the arrival of the conveyors transporting the products, which explains why energy consumption is null in this phase. As soon as the products arrive, the machines start assembling. The energy peak goes up to 800 watts and reaches 1200 watts.

In Figure 5, the consumption threshold is set to 800 watts. After the first phase of product waiting, the assembly of products is launched by the three machines (2, 3 and 4). The energy peak control algorithm prevents the imposed threshold from being exceeded (threshold = 800 watts). The production time is relatively long compared to the first solution, in which the energy threshold is unlimited.

Figure 4. Energy Consumption and Gantt chart without threshold for the manufacturing order 2xAIP

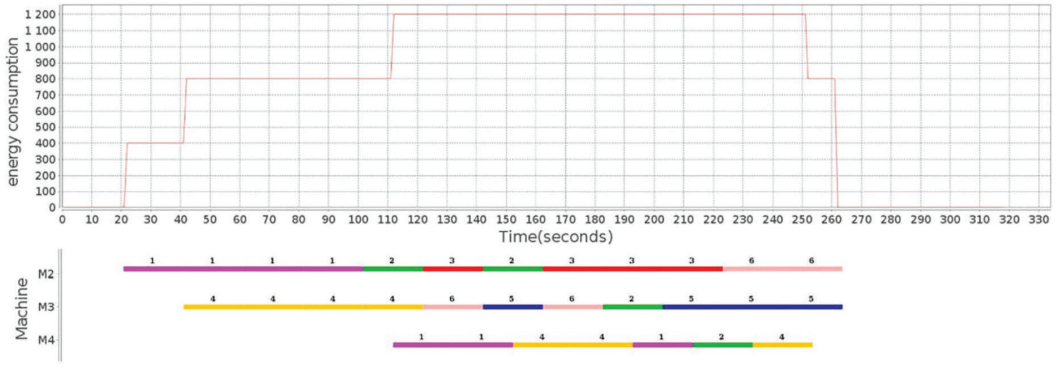
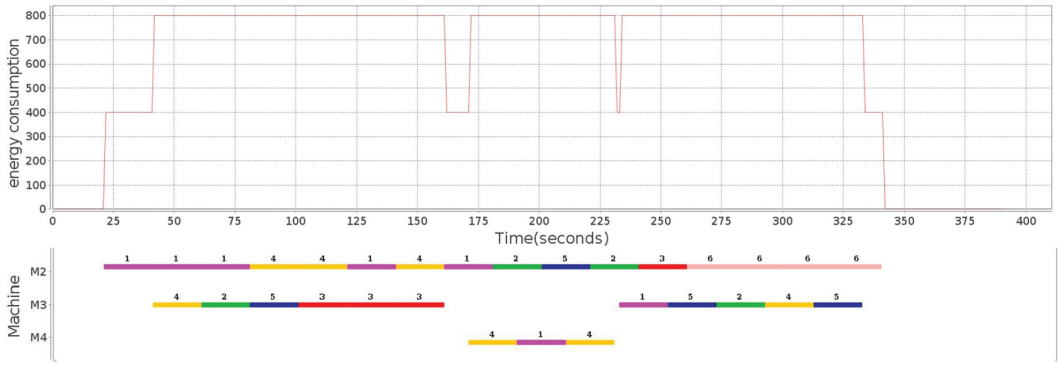


Figure 5. Energy consumption and Gantt chart with a threshold = 800 watts for the manufacturing order 2xAIP



In Figure 6, the energy threshold is set to 400 watts. The energy peak is stable during the assembly of the products and the distribution of the operations on the three machines is relatively balanced. However, the production time of this solution is greater compared to the previous solutions.

In the next experiment (Figure 7), the reactivity of the approach is studied with a dynamic threshold. Initially, the energy threshold is set to 800 watt then, it drops to 0 watts. In a third phase, it goes up to 800 watts and drops to 400 watts and finally it is set to 800 watts. The energy control algorithm reacts whenever a new threshold is imposed. This scenario is recurrent for example, when the energy supplier increases the price of energy in a given period.

Figure 8 shows a comparison between three scenarios of manufacturing process: without energy threshold, scheduling with energy threshold = 400 watts and 800 watts. According to this experience, the relationship between the energy threshold and the makespan is conflicting.

(Pach, Berger, Sallez, & Trentesaux, 2015) have proposed a reactive approach for energy control, based on the Potential Fields technique. They applied this approach to the same production cell (AIP-PREMICA).

Table 2 presents a comparison between the approach of (Pach, Berger, Sallez, & Trentesaux, 2015) and our approach (NSGA-2-ECA: Non-dominant Sorting Genetic Algorithm-Energy Control Algorithm). The production time is reduced to 15% for a threshold of 800 watts and 4.5% for a threshold of 400 watts. These results prove that the predictive-reactive approach is more powerful than pure reactive approach for dynamic energy control during manufacturing processes.

Figure 6. Energy Consumption and Gantt chart with a threshold = 400 watts for the manufacturing order 2xAIP

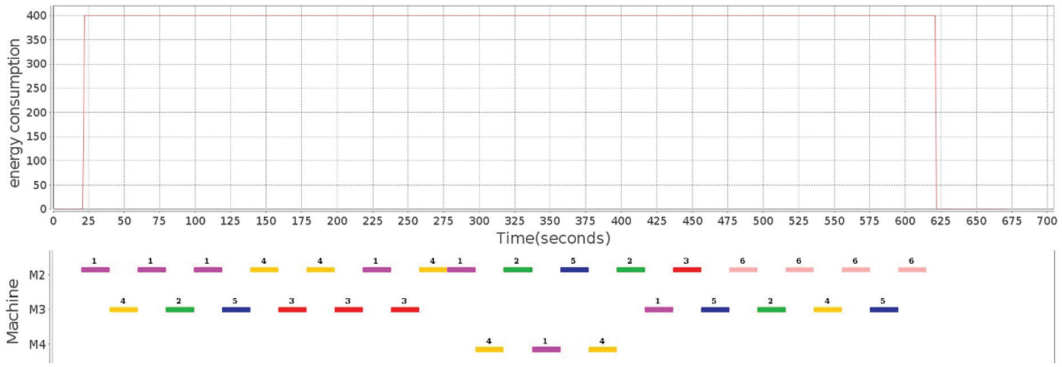


Figure 7. Energy consumption with dynamic thresholds for the manufacturing order 2xAIP

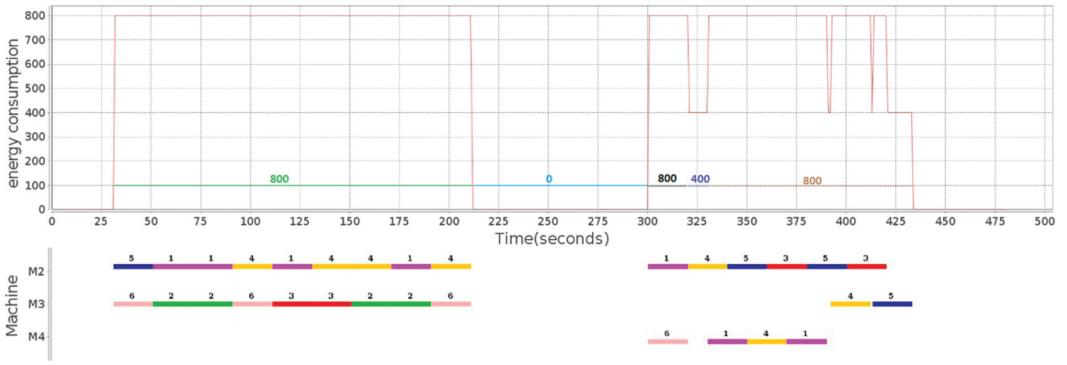


Figure 8. Manufacturing effectiveness with three scenarios: without threshold, energy threshold = 400 watts and 800 watts for the production order 2xAIP

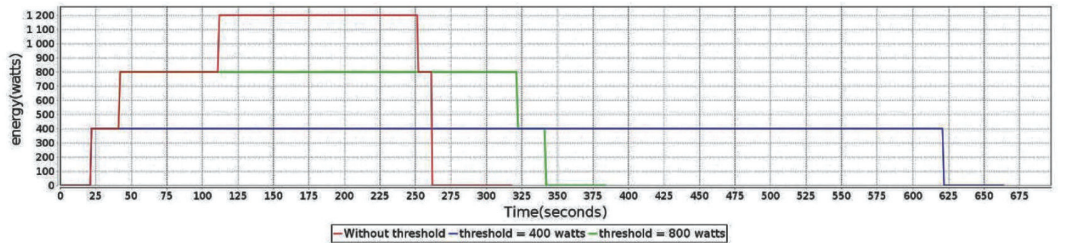


Table 2. Comparison between NSGA-2ECA and potential fields (Pach, Berger, Sallez, & Trentesaux, 2015)

	Potential Fields	NSGA-2-ECA*
Threshold of Power	**Cmax (secondes)**	
800 watts	400 s	**340 s**
400 watts	650 s	**621 s**

CONCLUSION

The purpose of the predictive-reactive approach is to discover new opportunities in efficiency of supply chain for industries, which involve many suppliers (intermediate manufacturers) to achieve a final product. Auto parts manufacturing is a typical example of these industries, where the process of car manufacturing is distributed across a supply chain, spanning multiple companies, and the overall business efficiency of manufacturing a car is impacted by each single supplier. This paper aims at reducing the cumulative energy consumption at the level of manufacturing processes of suppliers, which limit the total cost of the final product. The best balance between production time and energy consumption is found by multi-objective genetic algorithm. Consequently, companies can forecast the necessary energy and the production time to satisfy their customers' orders, which improve coordination between manufacturers and their customers. The reactive approach controls dynamically the energy consumption to face the uncertainty of energy volatility (e.g. energy decreasing, price increasing), and provides energy use data during manufacturing processes to make the changes

needed to reduce energy cost. This is useful to avoid downtime in intermediate companies (e.g. parts manufacturers) and optimize energy efficiency and agility in the supply chain.

The obtained results, compared with (Pach, Berger, Sallez, & Trentesaux, 2015), show that genetic algorithm is useful to predict energy volatility. This predictive method allows us to improve the production effectiveness (makespan) with energy efficiency. Consequently, the predictive-reactive approaches are more powerful than pure reactive approaches for energy volatility.

REFERENCES

Beasley, J. (1990). OR-library: Distributing test problems by electronic mail. *The Journal of the Operational Research Society*, *41*(11), 1069–1072. doi:10.1057/jors.1990.166

Belkaid, F., Yalaoui, F., & Sari, Z. (2016). An efficient approach for the reentrant parallel machines scheduling problem under consumable resources constraints. *International Journal of Information Systems and Supply Chain Management*, *9*(3). doi:10.4018/IJISSCM.2016070101

Boville, J., & Dussault, R. (2012). *Automotive energy management: finding hidden costs.* North Carolina: Schneider Electric.

Brandimarte, P. (1993). Routing and scheduling in a flexible job shop by tabu search. *Annals of Operations Research*, *41*(3), 157–183. doi:10.1007/BF02023073

Bruzzone, A., Anghinolfi, D., Paolluci, M., & Ronelli, F. (2012). Energy-aware scheduling for improving manufacturing process sustainability: A mathematical model for flexible flow shops. CIRP Annals, 61(1), 459-462.

Chen, J., Song, Z., Zheng, R., Zhao, F., & Yao, Z. (2010). *A Multi-objective Optimization Evolutionary Algorithm with Better Performances on Multiple Indicators. Computational Intelligence and Intelligent Systems. Communications in Computer and Information Science. 107*. Berlin: Springer.

Christian, A., Cyril, B., & Marie-Claude, P. (2002). Pilotage d'atelier basé sur un ordonnancement flexible. In P, Pujo & J.-P. Kieffer (Eds.), Méthodes du pilotage des systèmes de production (pp. 61-97). IC2 Productique.

Corne, D., Knowles, J. D., & Oates, M. J. (2000). The Pareto envelope based selection algorithm for multi-objective optimization. In International Conference on Parallel Problem Solving from Nature, LNCS (Vol. 1917, pp. 839-848). Springer. doi:10.1007/3-540-45356-3_82

Dai, M., Tang, D., Giret, A., Salido, M., & Li, W. (2013). Energy-efficient scheduling for a flexible flow shop using an improved genetic-simulated annealing algorithm. *Robotics and Computer-integrated Manufacturing*, *29*(5), 418–429. doi:10.1016/j.rcim.2013.04.001

Deb, K., Patrap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multi-objective genetic algorithm: NSGA-2. *IEEE Transactions on Evolutionary Computation*, *6*(2), 182–197. doi:10.1109/4235.996017

Fan, J., & Borlase, S. (2009). The Evolution of Distribution. *IEEE Power & Energy Magazine*, *7*(2), 63–68.

Fang, K., Uhan, N., & Sutherland, J. (2011). A new approach to scheduling in manufacturing in power consumption for power consumption and carbon footprint reduction. *Journal of Manufacturing Systems*, *30*(4), 234–240. doi:10.1016/j.jmsy.2011.08.004

Fang, K., Uhan, N. A., Zhao, F., & Sutherland, J. W. (2013). Flow shop scheduling with peak power consumption constraints. *Annals of Operations Research, 206(1), 115-145.* doi:10.1007/s10479-012-1294-z

Fonseca, C., & Fleming, P. (1993). Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. In *Proceedings of the Fifth International Conference on Genetic Algorithms (pp.* 416-423). San Francisco, CA: Morgan Kaufmann Publishers Inc.

Ghadimi, P., Kara, S., & Kornfeld, B. (2015). Renewable energy integration into factories: Real-time control of on-site energy systems. *CIRP Annals - Manufacturing Technology*, *64*(1), 443–446.

Ghasem, M., & Mehdi, M. (2011). A Pareto approach to multi-objective flexible job shop scheduling problem using particle swarm and local search. *International Journal of Production Economics*, *129*(1), 14–22. doi:10.1016/j.ijpe.2010.08.004

Giret, A., Trentesaux, D., & Prabhu, V. (2015). Sustainability in manufacturing operations scheduling: A state of art review. *Journal of Manufacturing Systems*, *37*, 126–140. www.elesivier.com/locate/jmasys doi:10.1016/j.jmsy.2015.08.002

Jensen, M. (2003). Reducing the run-time complexity of multiobjective EAs: The NSGA-2 and other algorithms. *IEEE Transactions on Evolutionary Computation*, *7*(5), 503-515. doi:10.110/TEVC.2003.817234

Jiang, Z., Zuo, L., & Mingcheng, E. (2014). Study on multi-objective flexible job-shop scheduling problem considering energy consumption. *Journal of Industrial Engineering and Management*, *7*(3), 589–604. doi:10.3926/jiem.1075

Katiraee, N., Shirazi, B., & Fazlollahtabar, H. (2017). Fuzzy Multi-Objective Supplier Selection considering Production Requirements in Resilient Supply Chain: Case Study in Steel Industry. *International Journal of Information Systems and Supply Chain Management*, *10*(3), 65–88. doi:10.4018/IJISSCM.2017070104

Lee, K. M., Yamakawa, T., & Lee, K. M. (1998). A genetic algorithm for general machine scheduling problems. In *1998 Second International Conference on Knowledge-Based Intelligent Electronic Systems, 1998. Proceedings KES'98.* (Vol. 2, pp. 60-66). IEEE.

Luo, H., Du, B., Huang, G., Chen, H., & Li, X. (2013). Hybrid flow shop scheduling considering machine electricity consumption cost. *International Journal of Production Economics*, *146*(2), 423–439. doi:10.1016/j.ijpe.2013.01.028

Marik, V., & McFarlane, D. (2005). Industrial adoption of agent-based technologies. *IEEE Intelligent Systems*, *20*(1), 27-35. Retrieved from http://doi.ieeecomputersociety.org/10.1109/MIS.2005.11

Pach, C., Berger, T., Sallez, Y., & Trentesaux, D. (2015). Reactive control of overall power consumption in flexible manufacturing systems scheduling: A Potential Fields model. *Control Engineering Practices*, *44*, 193-208. Retrieved from www.elsevier.com/locate/conengprac

Patch, C., Berger, T., Sallez, Y., Bonte, T., & Adam, E. (2014). Reactive and energy-aware scheduling of flexible manufacturing systems using potential fields. *Computers in Industry*, *65*(3), 434–448. doi:10.1016/j.compind.2013.11.008

Pouya, G., Sami, K., & Bernard, K. (2015). Renewable energy integration into factories: Real-time control of on-site energy systems. *CIRP Annals - Manufacturing Technology*, *64*(1), 443–446.

Prabhu, V. (2012). Services for Competitive and Sustainable Manufacturing in the Smart Grid in Service Orientation in Holonic and Multi-Agent Manufacturing Control. In T. Borangiu, A. Thomas & D. Trentesaux (Eds.), Service Orientation in Holonic and Multi-Agent Manufacturing Control (pp. 227-240). Springer.

Prabhu, V., & Taisch, M. (2012). Simulation modeling of energy dynamics in discrete manufacturing systems. In *Proceeding of the Information Control Problems in Manufacturing* (pp. 740-745). doi:10.3182/20120523-3-RO-2023.00422

Prabhu, v., Trentesaux, d., & Taisch, M. (2015). Energy-aware manufacturing operations. *International Journal of Production Research*, *53*(23).

Rajesh, R., Pugazhendhi, S., & Ganesh, K. (2011). Genetic Algorithm and Particle Swarm Optimization for Solving Balanced Allocation Problem of Third Party Logistics Providers. *International Journal of Information Systems and Supply Chain Management*, *4*(1), 24–44. doi:10.4018/jisscm.2011010102

Roghania, P., & Rasli, A. (2012). Productivity through effectiveness and efficiency in the banking industry. *Procedia: Social and Behavioral Sciences*, *40*, 550–556. doi:10.1016/j.sbspro.2012.03.229

Saaty, T.L. (2008). Decision making with the analytic hierarchy process. *International Journal of Services Sciences, 1*(1), 83-98.

Shrouf, F., Ordieres-Meré, J., García-Sánchez, A., & Ortega-Mier, M. (2014). Optimizing the production scheduling of a single machine to minimize total energy consumption costs. *Journal of Cleaner Production*, *67*, 197–207. doi:10.1016/j.jclepro.2013.12.024

Srinivas, N., & Deb, K. (1995). Multi-objective function optimization using non-dominated sorting genetic algorithm. *Evolutionary Computation*, *2*(3), 221–248. doi:10.1162/evco.1994.2.3.221

Taillard, E. (1993). Benchmarks for basic scheduling problem. *European Journal of Operational Research*, *64*(2), 278–285. doi:10.1016/0377-2217(93)90182-M

Tarek, C., Sondes, C., Nassima, A., & Damien, T. (2014). Scheduling under uncertainty: Survey and research directions. In *3rd International Conference on Advanced Logistics and Transport*, Hammamet, Tunisia. doi:10.1109/ICAdLT.2014.6866316

Trentesaux, D., Pach, C., Bekrar, A., Sallez, Y., Berger, T., Bonte, T., & Barbosa, J. et al. (2013). Benchmarking flexible job-shop scheduling and control systems. *Control Engineering Practice*, *21*(9), 1204–1225. doi:10.1016/j.conengprac.2013.05.004

Trentesaux, d., & Prabhu, V. (2014). Sustainability in manufacturing operations scheduling: stakes, approaches and trends. In *Proceedings of the IFIP WG 5.7 International Conference Advances in Production Management Systems* (pp. 106-113).

Wang, X., Ding, H., Qiu, M., & Dong, J. (2011). A low-carbone production scheduling systemconsidering renewable energy. In *Proceedings of IEEE International Conference on service operations, logistics and informatics* (pp. 101-106).

Xing, L., Chen, Y., & Yang, K. (2009). Multi-objective flexible job shop schedule: Design and evaluation by simulation modeling. *Applied Soft Computing*, *9*(1), 362–376. doi:10.1016/j.asoc.2008.04.013

Yildirim, M., & Mouzon, G. (2012). Single-Machine Sustainable Production Planning to Minimize Total Energy Consumption and Total Completion Time Using a Multiple Objective Genetic Algorithm. *IEEE Transactions on Engineering Management*, *59*(4), 585-597.

Zhang, L., Li, X., Zhang, G., & Wen, X. (2012). Dynamic scheduling model in FMS by considering energy consumption and schedule efficiency. In *Proceedings of the 2012 IEEE 16th International Conference on Computer Supported Cooperative Work in Design* (pp. 719-724). doi:10.1109/CSCWD.2012.6221898

ENDNOTES

1 https://www.iea.org/
2 www.strategyr.com
3 https://en.wikipedia.org/wiki/Argonne_National_Laboratory
4 http://www.univ-valenciennes.fr/bench4star/

Mohammed El Amine Meziane received Engineer and Magister degrees in computer science from the university of Oran 1 Ahmed Ben Bella (Algeria), 2008 and 2012. Researcher at the level of the laboratory LIO of the university of Oran 1 Ahmed ben Bella, Algeria. His main interest is operations research and optimization of the manufacturing systems using artificial intelligence.

Noria Taghezout is an assistant professor at university of Oran 1 Ahmed BenBella, Algeria. She holds her doctorate thesis in MITT at PAUL SABATIER UNIVERSITY in France in 2011.She also received another doctorate thesis in Distributed Artificial Intelligence from university of Oran 1 Ahmed BenBella in 2008. She holds a master's degree in Simulation and Computer aided-design. She conducts her research at the LIO laboratory as a chief of the research group in Modeling of enterprise process by using agents and WEB technologies. Since she studied in UPS Toulouse, she became a member of the EWG-DSS (Euro Working Group on Decision Support Systems). She is currently lecturing Collaborative decision making, Enterprise management and Interface human machine design. Her seminars, publications and regular involvement in Conferences, journals and industry projects highlight her main research interests in Artificial Intelligence.

Résumé

Nous proposons deux méthodes d'ordonnancement, destinées aux ateliers de production de type job shop flexible, en prenant en considération le transport entre machines. Un algorithme génétique amélioré par une recherche tabou avec une nouvelle fonction de voisinage pour minimiser la durée de production. Nous proposons ensuite, une méthode prédictive-réactive pour contrôler l'énergie consommée durant la production. Un algorithme génétique multi-objectifs (NSGA-2) avec trois objectifs (durée de production, consommation totale de l'énergie et le pic de consommation de l'énergie) est utilisé comme méthode prédictive. Un algorithme réactif de contrôle du pic de l'énergie consommée est proposé pour recalculer l'ordonnancement face à une exigence de baisse de consommation d'énergie. L'approche a été testée sur des benchmarks de la cellule de production AIP-PRIMECA, développée à l'université de Valenciennes en France. Les résultats ont été comparés avec d'autres études sur AIP-PREMICA.

Mots clés :

Systèmes manufacturiers flexibles; Optimisation mono objectif; Optimisation multi-objectifs; Optimisation de temps de production; Contrôle d'énergie; Contrôle de perte d'énergie; Méthode prédictive-réactive; NSGA-2; Ordonnancement du Job-shop flexible; Cellule de production AIP-PREMICA.