REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

Université Mohamed Boudiaf de M'sila
Faculté des Mathématiques et de l'Informatique
Département de Mathématiques

Mémoire de Master

Domaine : Mathématiques et Informatique
Filière : Mathématiques
Option : Analyse mathématiques et numérique

Thème

Étude d'un problème d'ordonnancement a machines parallèles identiques

Présenté par :
ELBAGGOUR Messaoud

Devant le jury composé de :

M^r *GAGUI* Bachir	grade,(MCA)	Université de M'sila	**Président**.
M^r *SELT* Omar	grade,(MCA)	Université de M'sila	**Encadreur**.
M^r *DILMI* Mustapha	grade,(MCB)	Université de M'sila	**Examinateur**.

Année universitaire 2020/2021

Dédicace

Je dédie ce modeste travail à : Mes chers parents,Abd Alah et Houria, qui m'ont donné tout le courage, la patience et la tendresse. Mes frères, ma soeur et toute ma famille. Mes amis et toute personne qui m'a aidé de loin et de près. Ma promotion de Master. Et sans oublier mes étudiants de Département de mathématique et l'Informatique.

Remerciements

Au nom d'Allah *le miséricordieux*

Ce mémoire marque une étape très importante dans ma vie couronnant mes années d'études universitaires, je voudrais remercier ici les personnes qui m'ont accompagné au cours de mon parcours, et qui ont participé de prés ou de loin à la réalisation de ce projet Je remercie avant tout Allah, tout puissant pour la volonté, la santé et la patience qu'il j'ai donnée durant toutes ces langues années d'études afin que je peux arriver à ce stade En second lieu, je tiens à remercier le professeur SELT Omar mon encadreur pour son encadrement, disponibilité et patience, Aussi je les remercie pour son encouragement, conseils et orientations et pour son soutien tout au long de cette période Un grand Mercie à mes parents qui m'ont données la chance de pour suivre mes études et qui m'ont appris à surpasser les moments difficiles .

♡

❖ *Messoaud*

Table des matières

1 Les problèmes d'ordonnancements **6**

1.1 Les éléments d'un problème d'ordonnancement 6

1.2 Les objectifs de l'ordonnancement 9

1.3 La typologie des problèmes d'ordonnancement 10

1.4 Résolution d'un Problème d'ordonnancement 10

1.5 Représentation des problèmes d'ordonnancement 11

1.5.1 Le diagramme de Gantt 11

1.5.2 Graphe potentiel-tâches 12

1.5.3 Method PERT (Program Evaluation and Research Task) 13

2 méthode de résolution des problémes d'ordonnancement **14**

2.1 Les méthodes d'optimisation 14

2.1.1 Les méthodes exactes 14

2.1.2 Les méthodes approchés 15

2.1.3 Les méthodes de descente (DM : Descent method) 18

3 quelques types des problèmes d'ordonnancement avec des exemples **25**

3.1 Modèles à une opération 25

3.1.1 Modèle à machine unique 25

3.1.2 Modèle à machines parallèle 25

3.2 Les problèmes d'ordonnancement d'atelier 30

3.3 Les types des Problèmes 31

3.3.1 Modèles à une opération 31

3.3.2 Modèles à plusieurs opérations 33

3.3.3 Modèle job-shop 34

Introduction Générale

Un problème d'ordonnancement est composé de façon générale d'un ensemble de tâches soumises à certaines contraintes, et dont l'exécution nécessite des ressources. Résoudre un problème d'ordonnancement consiste à organiser ces tâches, c'est-à-dire à déterminer leurs dates de démarrage et d'achèvement, et à leur attribuer des ressources, de telle sorte que les contraintes soient respectées. Les problèmes d'ordonnancement sont très variés. Ils sont caractérisés par un grand nombre de paramètres relatifs aux tâches (morcelables ou non, indépendantes ou non, durées fixes ou non), aux ressources (renouvelable sou consommables), aux types de contraintes portant sur les tâches (contraintes temporelles, fenêtres de temps . . .), aux critères d'optimalité liés au temps (délai total, délai moyen, retards . . .), aux ressources (quantité utilisée, taux d'occupation . . .) ou à d'autres coûts (production, lancement, stockage . . .).

Parmi tous ces problèmes, nous nous intéresserons à l'optimisation de critères réguliers pour des problèmes d'atelier et de fournées classés NP-Difficiles.

Dans un problème d'atelier, une pièce doit être usinée ou assemblée sur différentes machines. Chaque machine est une ressource disjonctive, c'est-à-dire qu'elle ne peut exécuter qu'une tâche à la fois, et les tâches sont liées exclusivement par des contraintes l'enchaînement. Plus précisément, les tâches sont regroupées en « n » entités appelées travaux ou lots. Chaque lot est constitué de « n » tâches à exécuter sur « m » machines distinctes.

Il existe Une classification très répandue des ateliers, du point de vue ordonnancement, est basée sur les différentes configurations des machines. Les modèles les plus connus sont ceux d'une machine unique, de machines parallèles, d'un atelier à cheminement unique (flow shop) ou d'un atelier à cheminement multiple (job shop). Un critère d'optimalité souvent étudié est la minimisation du délai total de l'ordonnancement (makespan).

Ce mémoire est composé de cinq chapitres dont nous présentons une brève description dans les paragraphes suivants :

Le premier chapitre présente les notions de base relatives aux problèmes d'optimisation combinatoire. Il rappelle aussi quelques concepts sur la théorie de la complexité, et donne un aperçu des éléments de la résolution des problèmes d'ordonnancement, leurs notations et leurs classifications .

Le deuxième chapitre de cette thèse, nous études les problèmes d'ordonnancement, voyons les différent domaines, puis nous verrons les différents éléments , la classification et la représentation du problème d'ordonnancement.

Le troisième chapitre nous allons expliquer les problèmes d'atelier et présent leur schéma de classification, expliqué les types de problèmes d'ordonnancement d'ateliers et la relation entre eux ,et leurs complexité.

Ensuite en le quatrième chapitre on explique seulement sur les problèmes de Machine parallèle et nous savoir les des méthodes de représentation de ces types de problème , suivi d'une description du problème à étudier.

Le cinquième et dernier chapitre portera sur l'implémentation de notre application ainsi que l'élaboration des tests expérimentaux.

Nous finirons notre travail par une conclusion générale qui présente un résumé de ce qu'a été étudié dans ce mémoire, les résultats obtenus et le travail qui reste à faire pour l'accomplissement de cette étude.

Chapitre 1

Les problèmes d'ordonnancements

Un problème d'ordonnancement consiste à organiser dans le temps la réalisation d'un en- semble de tâches, compte tenu de contraintes temporelles (délais, contraintes d'enchaînement, etc.), et de contraintes portant sur l'utilisation et la disponibilité des ressources requises pour les tâches, et visant à minimiser (resp. maximiser) un certain critère d'optimalité .

Dans ce chapitre on va vous présenter les différents éléments d'un problème de ordonnancement, leurs objectifs, leurs classifications . Nous verrons également comment résoudre et représentation des problèmes d'ordonnancement.

Définition 1.1. *Un problème d'ordonnancement peut être considéré comme un sous problème de planification dans lequel il s'agit de décider de l'exécution opérationnelle des tâches (jobs) planifiées, et ainsi d'établir leur planning d'exécution et leur allouer des ressources visant à satisfaire un ou plusieurs objectifs sous une ou plusieurs contraintes.*

En se basant sur les concepts de tâche, ressource, contrainte et objectif, l'ordonnancement peut également être défini comme :
« Ordonnancer un ensemble de tâches, c'est programmer leur exécution en leur allouant les ressources requises et en fixant leur date
de début ». [16]

1.1 Les éléments d'un problème d'ordonnancement

Dans un problème d'ordonnancement, quatre notions fondamentales interviennent : les tâches, les ressources, les contraintes et les objectifs. Dans ce qui suit, on donnera une définition détaillée de chacun de ces notions.

Les tâches.

Une tâche est une entité élémentaire de travail localisée dans le temps par une date de début et une date de fin d'exécution et qui consomme des ressources avec des quantités déterminées. Un coût (ou poids) est attribué à une tâche pour estimer sa priorité, son degré d'urgence, ou son coût d'immobilisation dans les système. On distingue deux types de tâches :

1. les tâches morcelables (préemptives) qui peuvent être exécutées en plusieurs fois, facilitant ainsi la résolution de certains problèmes.
2. les tâches non morcelables (indivisibles) qui doivent être exécutées en une seule fois et ne sont interrompues qu'une fois terminées. On note en général $N = \{J_1, J_2, \ldots, J_n\}$ l'ensemble des tâches, chaque tâche est caractérisée par :
3. La durée opératoire de la tâche i sur la machine j : (pij).
4. La date de disponibilité de la tâche i : (ri).
5. La date de début d'exécution de la tâche i : (si).
6. La date de fin d'exécution de la tâche i : (Ci).
7. La date d'achèvement souhaitée de la tâche i : (di).
8. Le facteur de priorité ou poids de la tâche i : (wi).
9. Le retard algébrique de la tâche i : $(Li = Ci - di)$.
10. Le retard vrai de la tâche i : $(Ti = max(Ci - di, 0))$.
11. L'indicateur de retard de la tâche i : $(Ui = 1, siTi > 0, Ui = 0$, sinon.

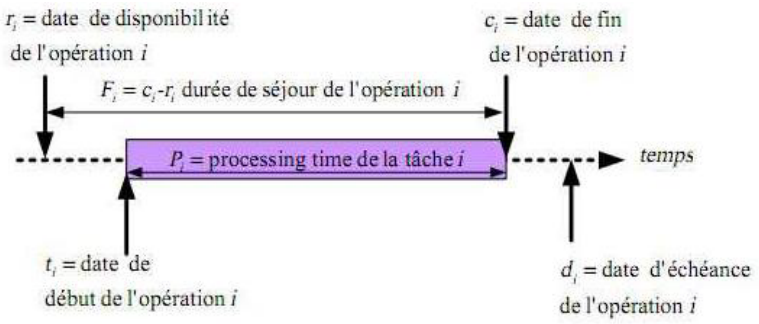

FIGURE 1.1 – La caractéristique d'une tache

Les ressources.

Une ressource est un moyen technique ou humain utilisé pour la réalisation d'une tâche et disponible en quantité limitée. Dans un atelier, plusieurs types de ressources sont distingués.

⁕ *Selon leurs disponibilités au cours du temps, on trouve :*

1. Les ressources renouvelables, comme c'est le cas pour les machines, personnels, équipements, etc. La ressource est dite renouvelable si après avoir été utilisé par une ou plusieurs opérations, elle est à nouveau disponible en même quantité.
2. Les ressources non-renouvelables, souvent appelées ressources consommables ou bien ressources financières. On dit que la ressource est non-renouvelables si ça disponibilité décroît après avoir été allouée à une opération. C'est le cas pour la matière première, budget.
3. Les ressources doublement contraintes, ces ressources combinent les contraintes liées aux deux catégories précédentes. Leur utilisation instantanées et leur consommation globale sont toutes les deux limitées. C'est le cas des ressources d'énergie (pétrole, électricité, etc.).

⁕ *Selon leurs capacités, on trouve :*

1. Les ressources disjonctives (ou non-partageables), il s'agit des ressources qui ne peuvent exécuter qu'une seule opération à la fois c'est le cas par exemple de machineoutil ou robot manipulateur.
2. Les ressources cumulatives (ou partageables), il s'agit des ressources qui peuvent être utilisé par plusieurs opérations simultanément (équipes d'ouvriers, poste de travail, etc.).[11]

Les contraintes.

Les contraintes expriment des restrictions sur les valeurs que peuvent prendre conjointement les variables de décision 1 . En d'autres termes, les contraintes représentent les

conditions à respecter lors de la construction de l'ordonnancement pour qu'il soit réalisable. Plus les contraintes sont nombreuses, plus le problème d'ordonnancement devient plus difficile. .[11].

Suivant la disponibilité des ressources et suivant l'évolution temporelle, deux types de contraintes peuvent être distingués : contraintes de ressources et contraintes temporelles

- **les contraintes de ressources** :

plusieurs types de contraintes peuvent être induits parla nature des ressources. A titre d'exemple, la capacité limitée d'une ressource implique un certain nombre, à ne pas dépasser, de tâches à exécuter sur cette ressource.

Les contraintes relatives aux ressources peuvent être disjonctives, induisant une contrainte de réalisation des tâches sur des intervalles temporels disjoints pour une même ressource, ou cumulatives impliquant la limitation du nombre de tâches à réaliser en parallèle. • les contraintes temporelles :

elles représentent des restrictions sur les valeurs que peuvent prendre certaines variables

1. d'ordonnancement. Ces contraintes peuvent être :

2. des contraintes de dates butoirs, certaines tâches doivent être achevées avant une date préalablement fixée, des contraintes de précédence, une tâche i doit précéder la tâche j
3. des contraintes de dates au plus tôt, liées à l'indisponibilité de certains facteurs nécessaires pour commencer l'exécution des tâches.

1.2 Les objectifs de l'ordonnancement

Les objectifs dits aussi les critères d'évaluation sont les indicateurs de performance sur lesquels se base le choix d'un ordonnancement satisfaisant. En ordonnancement, les

critères à optimiser consistent à minimiser ou maximiser une fonction objectif. Cette fonction objectif est générale- ment liée aux temps, aux ressources ou bien aux coûts. [11].

Les objectifs liés au temps :

L'ordonnancement joue sur la classification des tâches pour minimiser le temps total d'exécution, du temps moyen d'achèvement, des durées totales de règles ou des retards par rapport aux dates de livraison.

Les objectifs liés aux ressources :

L'ordonnancement pour objectifs d'optimisation l'utilisation des ressources ou le nombre de ressource nécessaires pour réaliser un ensemble de tâches. . . .etc .

Les objectifs liés au coût :

Ces objectifs sont généralement de minimiser les couts, de lancement, de production, de stockage, de transport, etc.

1.3 La typologie des problèmes d'ordonnancement

Une typologie des problèmes d'ordonnancement dans un atelier peut s'opérer selon le nombre et la nature des machines ainsi que l'ordre d'enchaînement des opérations (gamme defabrication). Deux grandes familles de problèmes d'ordonnancement se présentent. La première famille regroupe les problèmes pour lesquels chaque job nécessite une seule opération. La deuxième regroupe ceux dont les jobs nécessitent plusieurs opérations.

1.4 Résolution d'un Problème d'ordonnancement

Résoudre un problème d'ordonnancement, c'est choisir pour chaque tache une date de début, de telle sorte que les contraintes du problème soient respectées (la solution est alors dite admissible ou réalisable) et qu'un ou plusieurs critères donnés soient optimisés. La résolution d'un problème d'ordonnancement doit concilier deux objectifs :

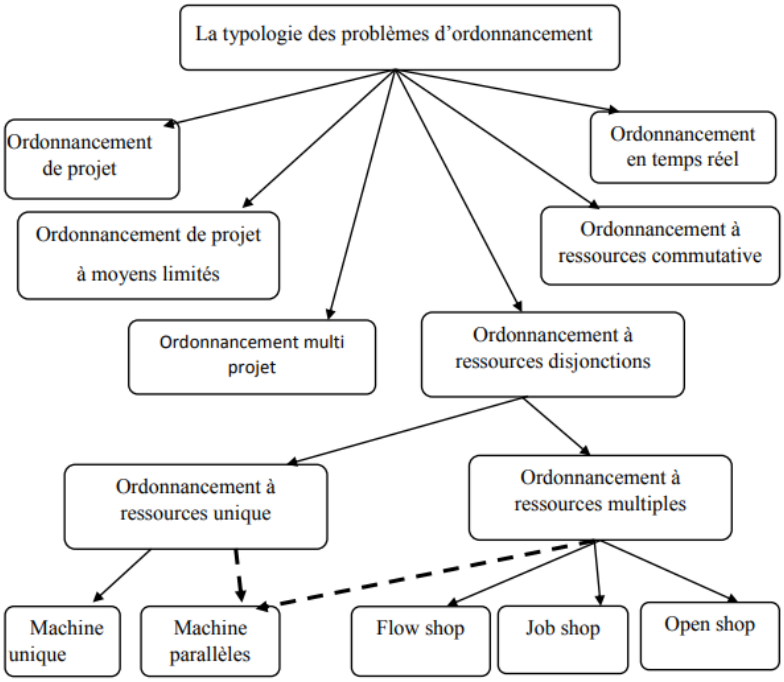

FIGURE 1.2 –

1. L'aspect statique consiste à générer un plan de réalisation des travaux sur la base des données prévisionnelles.
2. L'aspect dynamique consiste à prendre des décisions en temps réel, compte tenu de l'état des ressources et l'avancement dans le temps des différentes taches.

1.5 Représentation des problèmes d'ordonnancement

Il existe des sortes de représentations possibles d'un problème d'ordonnancement, le diagramme de Gantt, le graphe Potentiel-Tâches et la méthode PERT.

1.5.1 Le diagramme de Gantt

La représentation par Le diagramme de Gantt est la plus courante pour l'ordonnancement. Celui-ci représente une opération par un segment ou une barre horizontale, dont la longueur est proportionnelle à sa durée opératoire.

Donc, sur ce diagramme sont indiqués, selon une échelle temporelle : l'occupation des machines par les différentes tâches, les temps morts et les éventuelles indisponibilités des machinesdues aux changements entre produits. [19]

Deux types de diagramme de Gantt sont utilisés : Gantt ressources et Gantt tâches (voir la Figure 2.3)

Le diagramme de Gantt ressource, est composé d'une ligne horizontale pour chaque ressource (machine). Sur cette ligne, sont visualisées les périodes d'exécution des différentes opérations en séquence et les périodes de l'oisiveté de la ressource.

Le diagramme de Gantt tâches permet de visualiser les séquences des opérations des tâches, en représentant chaque tâche par une ligne sur laquelle sont visibles, les périodes d'exécution des opérations et les périodes ou la tâche est en attente des ressources.

1.5.2 Graphe potentiel-tâches

Cette outil graphique a été développé grâce à la théorie des réseaux de Pétri qui ont surtout servi à modéliser les systèmes dynamiques à évènements discrets.

Dans ce genre de modélisation, les tâches sont représentées par des noeuds et les contraintes par des arcs, comme le montre la (Figure 2.4). Ainsi, les arcs peuvent être de deux types

1. les arcs conjonctifs illustrant les contraintes de précédence et indiquant les durées des tâches.
2. les arcs disjonctifs indiquant les contraintes de ressources.

Dans l'exemple représenté dans « Figure 4.2 » la tâche « f » ne peut s'exécuter que si a et d ont été réalisées. Donc, pour exécuter « a » il faut 6 mois et pour exécuter « d » il faut 2+3 mois. La tâche « f » ne pourra commencer au plus tôt que 6 mois après le début du projet : c'est donc le plus long chemin entre « a » et « f ».

- la durée du projet, qui correspond au plus long chemin entre S (tâche de début du projet) et S' (tâche de fin du projet).

1.5.3 Method PERT (Program Evaluation and Research Task)

Cette représentation, semblable à la précédente, permet de représenter une tâche par un arc, auquel est associé un chiffre qui représente la durée de la tâche. Entre les arcs, figurent des cercles, appelés sommets ou événements, qui marquent l'aboutissement d'une ou de plusieurs tâches. Ces cercles sont numérotés afin de suivre l'ordre de succession des divers évènements. Les méthodes graphiques ont connu une très importante évolution surtout avec l'apparition des Réseaux de Pétri, qui permettent de traduire plusieurs notions fondamentales ayant un lien avec les problèmes d'ordonnancement, telles que :

1. Les conflits sur les ressources.
2. Les durées opératoires certaines.
3. Les durées opératoires aléatoires.
4. Les gammes.
5. Les disponibilités, les multiplicités et les capacités de ressources la répétitivité, etc

Chapitre 2

méthode de résolution des problémes d'ordonnancement

2.1 Les méthodes d'optimisation

Comme on l' rappelé dans le paragraphe sur la complexité, la plupart des problèmes d'ordonnancement sont NP-difficiles.On ne peut donc pas espérer trouver un ordonnancement optimal en un temps raisonnable pour des problèmes de taille industrielle. Les méthodes utilisées sont donc des méthodes approchées.

Pour des problèmes de petite taille, nous pouvons obtenir une solution exacte.Une méthode exacte peut servir pour résoudre des sous-problèmes d'un problème de grande taille de manière optimale.

2.1.1 Les méthodes exactes

Les méthodes exactes utilisent surtout deux approches de résolution très connues : la programmation dynamique et les procédures par séparation et évaluation. Ces méthodes sont souvent utilisées pour résoudre les problèmes combinatoires de manière exacte, en ordonnancement tout particulièrement. Ce sont des méthodes d'énumération implicite.

L'énumération explicite construit toutes les solutions réalisables et retient une parmi les meilleures.L'énumération implicite consiste à explorer l'ensemble de toutes les solutions réalisables en éliminant des sous-ensembles de solutions moins intéressants sans avoir à les construire. Nous pouvons citer trois approches particulièrement célèbres : La programmation dynamique introduite par Bellman dans les années 504 , la méthode par séparation et évaluation (Branch and Bound en anglais : notée B et B) et l'algorithme de retour arrière (Backtracking).

2.1.2 Les méthodes approchés

Une méthode approchée est une méthode d'optimisation qui a pour but de trouver une solution réalisable de la fonction objectif en un temps raisonnable, mais sans garantie d'optimisation L'avantage principal de ces méthodes est qu'elles peuvent s'appliquer à n'importe quelle classe de problèmes, faciles ou très difficiles, d'un autre côté les algorithmes d'optimisation que les algorithmes de recuit simulé, les algorithmes tabous et les algorithmes génétiques ont démontré leurs robustesses et efficacités face à plusieurs problèmes d'optimisation combinatoires.

Les méthodes approchées englobent deux classes : les heuristiques et les métaeu- ristiques. La particularité qui différencie les méthodes métaheuristiques des méthodes heuristiques c'est que les métaheuristiques sont applicables sur de nombreux problèmes. Tandis que, les heuristiques sont spécifiques à un problème donné.

Les heuristiques :.

Les méthodes heuristiques sont des méthodes spécifiques à un problème particulier. Elles nécessitent des connaissances du domaine du problème traité. En fait, se sont des règles empiriques qui se basent sur l'expérience et les résultats acquis afin d'améliorer les recherches futures. Plusieurs définitions des heuristiques ont été proposées par plusieurs chercheurs dans la littérature, parmi les quelles :

Définition 2.1. *Une heuristique (règle heuristique, méthode heuristique) est une règle d'estimation, une stratégie, une astuce, une simplification, ou tout autre type de dispositif qui limite considérablement la recherche de solutions dans des espaces problématiques importants. Lesheuristiques ne garantissent pas des solutions optimales. En fait, elles ne garantissent pas une solution du tout. Tout ce qui peut être dit d'une heuristique utile, c'est qu'elle propose des solutions qui sont assez bonnes la plupart du temps.*

Définition 2.2. *Une méthode heuristique (ou simplement une heuristique) est une méthode qui aide à découvrir la solution d'un problème en faisant des conjectures plausibles mais faillible de ce qui est la meilleure chose à faire.*

Définition 2.3. *Les heuristiques sont des ensembles de règles empiriques ou des stratégies qui fonctionnent, en et effet, comme des règles d'estimation.*

Les métaheuristiques :.

Les métaheuristiques d'optimisation sont des algorithmes généraux d'optimisation applicables à une gran de variété de problèmes.Elles sont apparues à partir des années 80, dans le but de résoudre au mieux des problèmes d'optimisation. Les métaheuristiques s'efforcent de résoudre tout type de problème d'optimisation. Elles sont caractérisées par leur caractère stochastique.Ainsi que par leur origine discrète.

Elles sont inspirées par desanalogies avec la physique (recuit simulé, recuit microcanonique), avec labiologie (algorithmes évolutionnaires) ou encore l'éthologie (colonies defourmis, essaims particulaires). Cependant, elles ont l'inconvénient d'avoir plusieurs paramètres à régler. Il est à souligner que les métaheuristiques se prêtent à toutes sortes d'extensions, notamment en optimisation mono-objectif et multi-objectif.

Les métaheuristiques utilisent des recherches stratégiques affin d'explorer plus efficacement l'espace de recherche, et souvent se focalisent sur les régions prometteuses. Ces méthodes commencent par un ensemble de solutions initiales ou une population initiale, et après, elles examinent étape par étape une séquence de solutions pour atteindre, oude s'approcher de la solution optimale du problème. Les métaheuristiques ont plusieursavantages par rapport aux algorithmes traditionnels. Les deux avantages les plus importants sont la simplicité et la flexibilité. Les métaheuristiques sont souvent simples à implémenter, pourtant, elles sont capables de résoudre des problèmes complexes avec la capacité de s'adapter à plusieurs problèmes d'optimisation du monde réel, à partir du domaine de la recherche opérationnelle, d'ingénierie vers l'intelligence artificielle.

De nos jours les gestionnaires et les décideurs sontconfrontés quotidiennement à des problèmes de complexité grandissante,qui surgissent dans des secteurs très divers. Le problème à résoudre peutsouvent s'exprimer sous la forme générale d'un problème d'optimisation, dans lequeL.

on définit une ou plusieurs fonctions objectif que l'on cherche à minimiser ou à maximiser par rapport à tous les paramètres concernés.

Le mot métaheuristique est dérivé de la composition de deux mots grecs :

1. heuristique qui vient du verbe heuriskein et qui signifie "trouver"

2. méta qui est un suffixe signifiant au-delà "dans un niveau supérieur"

Les propriétés fondamentales des métaheuristiques :

1. Les métaheuristiques sont des stratégies qui permettent de guider la recherche d'une solution optimale.
2. Le but visé par les métaheuristiques est d'explorer l'espace de recherche efficacement afin de déterminer dessolutions (presque) optimales.
3. Les techniques qui constituent des algorithmes de type métaheuristique vont de la simple procédure derecherche locale à des processus d'apprentissage complexes.
4. Les métaheuristiques sont en général non-déterministes et ne donnent aucune garantie d'optimalité
5. Les métaheuristiques peuvent contenir des mécanismes qui permettent d'éviter d'*ê*tre bloqué dans des régionsde l'espace de recherche.
6. Les métaheuristiques peuvent faire appel à des heuristiques qui tiennent compte de la spécificité du problèmetraité, mais ces heuristiques sont contrôlées par une stratégie de niveau supérieur.
7. Les métaheuristiques peuvent faire usage de l'expérience accumulée durant la recherche de l'optimum, pour mieux guider la suite du processus de recherche

Classification des métaheuristiques :.

On peut regrouper les métaheuristiques en deux grandes classes : les métaheuristiques à solution unique (c.à.d. évoluant avec une seule solution) et celles à solutions multiples ou population de solutions. Les méthodes d'optimisation à population de solutions améliorent, au fûr et à mesure des itérations, une population de solutions. L'intérêt de ces méthodes est d'utiliser la population comme facteur de diversité.

Les métaheuristiques à solution unique :.

Dans cette section, nous présentons les métaheuristiques à base de solution unique, aussi appelées méthodes de trajectoire.Contrairement aux métaheuristiques à base de population, lesmétaheuristiques à solution unique commencent avec une seule solution

initiale et s'en éloignent progressivement, en construisant une trajectoire dans l'espace de recherche. Les méthodes detrajectoire englobent essentiellement la méthode de descente, le recuit simulé, la recherche tabou, la recherche à voisinage variable, la méthode GRASP, la recherche locale itérée, la recherche locale guidée et leurs variantes.

2.1.3 Les méthodes de descente (DM : Descent method)

.

Ces méthodes s'arti-culent toute sautour d'un principe simple. Partir d'une solution existante, chercher une solution dans le voisinage et accepter cette solution si elle améliore la solution courante.

L'algorithme 1 présente le squelette d'une méthode de descente simple (Simple descent). Apartir d'une solution initiale x on choisit une solution x' dans le voisinage $N(x)$ de x Si cette solution est meilleure que $xf(x') < f(x)$ alors on accepte cette solution comme nouvelle solution x et on recommence le processus jusqu'à ce qu'il n'y ait plus aucune solution améliorante dans le voisinage de x
algorithme :1 simple descente

1 : initialise :find an initial solution X

2 : repeat

3 : neighbourhood search : find a solution x'

4 : if $f(x') < f(x)$ then

5 : $x \longrightarrow x'$

6 : end if

7 : until $f(y) \geq f(x); \quad \forall y \in N(x)$

Une version plus agressive de la méthode de descente est la méthode deplus grande descente (Deepest descent). Au lieu de choisir une solution x' dans le voisinage de x on choisit toujours la meilleure solution x' du voisinage de x
l'algorithme 2 : donne une description de cette méthode.

algorithme 2 : deepest descente

1 : initialise : find an initial solution x

2 : repeat

3 : neighbourhood search : find a solution $x' \in N(x)/f(x') \leq f(x''), \forall x'' \in N(x)$

4 : if $f(x') < f(x)$ then

5 : $x \longrightarrow x'$

6 : end if

7 : until $f(x') \geq f(x), \quad \forall x' \in N(x)$

Ces deux méthodes sont évidemment sujettes a de nombreuses critiques. Ellesse basent toutes les deux sur une amélioration progressive de la solution et doncresteront bloquées dans un minimum local des qu'elles en rencontreront un. Il existe de manière évidente une absence de diversification. L'équilibre souhaité entre intensification et diversification n'existe donc plus et l'utilisateur de ces deux méthodes doit en être conscient.

Un moyen très simple de diversifier la recherche peut consister à re-exécuterun des algorithmes en prenant un autre point de départ. Comme l'exécution de ces méthodes est souvent très rapide, on peut alors inclure cette répétition au sein d'une boucle générale. On obtient alors un algorithme de type "Multi-start descent" décrit par l'algorithme 3.

algorithme3 : multi-start descente :

1 : initialise :find an initial solutin $1 \longrightarrow k, x; +1 \longrightarrow f(B)$

2 : repeat

3 : starting point :choose an initial solution x_0 at random

4 : $\longrightarrow x$ result of simple Descent or Deepest Descent

5 : if $f(x) < f(B)$ then

6 : $x \longrightarrow B$

7 : end if

8 : $k+1 \longrightarrow k$

9 : unitl stopping criterion satisfied

De manière évidente, la diversification est totalement absente des algorithmes 1 et 2. En ne conservant que l'aspect intensification,la convergence est souvent trop rapide et on se trouve très rapidement bloqué dans un optimum local. Les résultats communément admis indiquent que ces techniques conduisent engénéral à des solutions en moyenne à 20% de l'optimum. Dans le cas de l'algorithme 3, la diversification est simplement insérée par le choix aléatoire d'une solution de départ

La recherche tabou (TS : Tabu Search) :.

La recherche tabou est une méta- heuristique originalement développée par Glover, 1986 et indépendamment par Hansen, 1986 . Elle est basée sur des idées simples, mais elle est néanmoins très efficace. Cette méthodecombine une procédure de recherche locale avec un certain nombre de règles et demécanismes permettant à celle-ci de surmonter l'obstacle des optima locaux, tout en évitant de cycler. Elle a été appliquée avec succès pour résoudre de nombreux problèmes difficiles d'optimisation combinatoire : problèmes de routage devéhicule, problèmes d'or- donnancement, problèmes de coloration de graphes, etc.

Dans une première phase, la méthode de recherche tabou peut être vuecomme une généralisation des méthodes d'amélioration locales. En effet, en partant d'une solution quelconque x appartenant à l'ensemble de solutions S, on se déplacevers une solution x' située dans le voisinage $N(x)$. Donc l'algorithme explore itérativement l'espace de solutions S. algorithme 4 :TS

1 : initialise :find an initial solution x

2 : repeat

3 : neighbourhood search :find a solution $x' \in N^*(x)$

4 : update memory : tabu list ,frequency-based memory, aspiration level,...

5 : move $x' \longrightarrow x$

6 : unitl stopping criterion statisffied

Afin de choisir le meilleur voisin x' dans $N(x)$, l'algorithme évalue lafonction objectif f en chaque point x' et retient le voisin qui améliore la valeur de la fonction objectif, ou au pire celui qui la dégrade le moins

Les métaheuristiques à population de solutions : .

Contrairement aux algorithmespartant d'une solution singulière, les métaheuristiques à population de solutions amé- liorent, au fur et à mesure des itérations, une population de solutions.On distingue dans cette catégorie, les algorithmes évolutionnaires, qui sont une famille d'algorithmes is- sus de la théorie de l'évolution par la sélection naturelle, énoncée par Charles Darwin et les algorithmes d'intelligence en essaim qui, de la même manièreque les algorithmes évolution naires, proviennent d'analogies avec des phénomènes biologiques naturels.

Les algorithmes génétiques (GA :Genetic Algorithm) :.

Proposé dans les années 1975 par Holland, les algorithmes génétiques doivent leur popularité à Goldberg. Avant la parution de son livre qui est une des références les plus citées dans le domaine de l'informatique, on a puvoir un certain nombre d'autres présentations, citons Goldberg , Holland, Schwefel Le sujet connaît une très grande popularité. Il existe au jour d'huiplu- sieurs milliers de références sur le sujet et le nombre de conférences dédiéesau domaine (que ce soit sur les techniques elles-mêmes ou sur les applications)ne fait qu'augmenter.

De manière générale, les algorithmes génétiques utilisent un même principe.Une population d'individus (correspondants à des solutions) évoluent en même temps comme dans l'évolution naturelle en biologie. Pour chacun des individus, on mesure sa faculté d'adaptation au milieu extérieur par le fitness.

Les algorithmes génétiques s'appuient alors sur trois fonctionnalités :

1. *La Sélection :* Pour déterminer quels individus sont plus enclins à se reproduire, une sélection est opérée. Il existe plusieurs techniques de sélection, les principales utilisées sont la sélection par tirage à la roulette (roulette-wheel selection), la sélection par tournoi (tournament selection), la sélection par rang (ranking selection).

2. *Le Croisement :* L'opérateur de croisement combine les caractéristiques d'un ensemble d'individus parents (généralement deux) préalablement sélectionnés, et génère de nouveaux individus enfants. Là encore, il existe de nombreux opérateurs de croisement, par exemple le croisement en un point, le croisement en n-points ($n \geq 2$) et le croisement uniforme 5.
3. *La Mutation et le remplacement :* Les descendants sont mutés, c'est-à-dire que l'on modifie aléatoirement une partie de leur génotype, selon l'opérateur de mutation.Le rem- placement (ou sélection des survivants), comme son nom l'indique, remplace certains des parents par certains des descendants. Le plus simple est de prendre les meilleurs indivi- dus de la population, en fonction de leurs performances respectives, afin de former une nouvelle population (typiquement de la même taille qu'au début de l'itération).

La représentation des solutions (le codage) est un point critique de la réussite d'un algorithme génétique. Il faut bien sûr qu'il s'adapte le mieux possible au problème et à l'évaluation d'une solution. Le codage phénotypique ou codage direct correspond en général à une représentation de la solution très proche de la réalité. L'évaluation d'une solution représentée ainsi est en général immédiate.

L'algorithme 3 propose une version qualifiée en Anglais par population replacement (ou algorithme générationnel)

algorithme 5 :GA

1 : initial : generate an initial population p of solution with size $|p| - n$

2 : repeat

3 : $\sigma \longrightarrow p'$

4 : repeat σ

5 : selection : choose 2 solution x and x' from p with propability proportional to their tness

6 : crossover :combine parent solution x and x' to from child solutions y and y' with high propability

7 : mutate y and y' with smal propability

8 : add y and y' to p'

9 : until $|p'| = n$

10 : $p' \longrightarrow p$

11 : until stopping criterion

Algorithme de colonies de fourmis (ACO : Ant Colony Optimization) :

Les algorithmes de colonies de fourmis ont été proposés par Colorni,Dorigo et Maniezzo en 1992 et appliquées la première fois au problème du voyageur de commerce. Ce sont des algorithmes itératifs à population où tous les individus partagent un savoir commun qui leur permet d'orienter leurs futurs choix et d'indiqueraux autres individus des choix à suivre ou à éviter. Le principe de cette métah euristique repose sur le comportement particulier des fourmis, elles utilisent pour communiquer une substance chimique volatile particulière appelée phéromone grâce à une glande située dans leur abdomen. Enquittant leur nid pour explorer leur environnement à la recherche dela nourriture (Food), les fourmis arrivent à élaborer des chemins qui s'avèrent fréquemment être les plus courts pour aller du nid vers une source de nourriture. Chaque fourmi dépose alors une quantité de phéromonessur ces pistes qui deviendront un moyen de communication avec leurscongénères, les fourmis choisissent ainsi avec une probabilité élevée les chemins contenant les plus fortes concentrations de phéromones à l'aide des récepteurs situés dans leurs
des figures illustre et confirme ce constat, une expérience a été faite par Gauss et Deneubourg en 1989 appelée expérience du pont à double branche, les fourmis qui re- tournent au nid rapidement, après avoir visité la source de nourriture, sont celles qui ont choisi la branche courteet les fourmis empruntant cette branche faisant plus d'aller retour, et par conséquent la quantité de phéromones déposée sur la plus courte branche est relativement supérieure que celle présente sur la plus longue branche. Puisque les fourmis sont attirées plus vers les pistes de plus grande concentration en phéromones, alors la branche courte sera la plus empruntée par la majorité des fourmis. antennes.

des figures illustre et confirme ce constat, une expérience a été faite par Gauss et Deneubourg en 1989 appelée expérience du pont à double branche, les fourmis qui retournent au nid rapidement, après avoir visité la source de nourriture, sont celles qui

ont choisi la branche courteet les fourmis empruntant cette branche faisant plus d'aller retour, et par conséquent la quantité de phéromones déposée sur la plus courte branche est relativement supérieure que celle présente sur la plus longue branche. Puisque les fourmis sont attirées plus vers les pistes de plus grande concentration en phéromones, alors la branche courte sera la plus empruntée par la majorité des fourmis.

algirithme 6 :ACO

1 : unitialise :create an initial population of ants

2 : repeat

3 : for each and do

4 : construct a solution based on the construction procedure , biased by the pheromone trails

5 : update the pheromone trails based on the quality of the solutions found

6 : end for

7 : until stopping criterion satisfied

Chapitre 3

quelques types des problèmes d'ordonnancement avec des exemples

Les problèmes d'ordonnancement sont généralement classés en deux principaux modè les dépendamment du nombre d'opérations que requièrent les jobs : des modèles à une opération (machine unique et machines parallèles) et des modèles à plusieurs opérations (flow-shop, open shop et job shop) dans ce chapitre on va expliquer le modéle à une opération .

3.1 Modèles à une opération

3.1.1 Modèle à machine unique

Dans un modèle à machine unique, l'ensemble des tâches à réaliser est exécuté par une seule machine.L'une des situations intéressantes où on peut rencontrer ce genre de configurations est devant un système de production comprenant une machine goulot qui influence l'ensemble du processus.

3.1.2 Modèle à machines parallèle

Ce modèle est utilisé surtout dans les secteurs industriels tels que : l'industrie alimentaire , les industries plastiques, les fonderies et en particulier l'industrie textile. Le processus de déroulement de ce système de production est le suivant : à chaque fois qu'une machine i se libère, ont lui affecte un job j . Dans le cas d'un processus d'assem-blage industriel,

Définition 3.1. *Le principe des problèmes d'ordonnancement à machines parallèles est Chaque job est constitué d'une seule opération et chaque opération peut être réalisée par n'importe laquelle des machines, disposées en parallèles, mais n'en nécessite qu'une seule. On consi- dère dans toute cette section le critère de minimisation de la durée totale. Dans le cadre de l'informatique on peut ainsi modéliser les « m » processeurs d'une machine parallèle. Ces problèmes sont presque toujours NP-difficiles.*

Makespan (cmax) :

Le makespan représente le temps de fin d'exécution du dernier job dans une séquence. Il est l'un des critères les plus utilisés pour évaluer le coût d'un ordonnancement. En minimisant ce critère, on peut améliorer le rendement et réduire le temps moyen d'inactivité des machines. La minimisation du makespan s'accompagne généralement de contraintes qui peuvent être temporelles ou liées aux ressources. Les contraintes temporelles se divisent en deux catégories : des contraintes de temps alloué (impératif de gestion : délai de livraison, disponibilité, achèvement) et des contraintes d'antériorité (cohérence technologique :gammes de fabrication,inégalité de potentiels :précédence).Les contraintes. liées aux ressources peuvent être des contraintes disjonctives (une tâche i doit s'exécuter avant ou après une tâche) ou des contraintes cumulatives (respect des capacités des ressources).

On peut aussi considérer d'autres critères de performance, tels que le temps moyen d'achèvement des jobs, le temps total de traitement, le temps de retard total, le temps d'attente des jobs, le taux d'occupation de machines, le nombre de jobs en retard, le temps de séjour d'un job dans le système avant sa réalisation, etc.

Les différents modèles :

Pour permettre la classification des problèmes d'ordonnancement, ceux-ci ont été séparés en deux grands modèles suivant les tâches à exécuter

1. *Les modèles dits statiques :* Le nombre de tâches à exécuter et leurs dates de disponibilité sont connus à l'avance et ne peuvent être modifiés par la suite,
2. *Les modèles dynamiques :* Les tâches arrivent de façon aléatoire. Leurs nombres, ainsi que leurs dates de disponibilité ne sont pas connus à l'avance.

Les modèles mathématique des problèmes à machine parallèle :

Après les études des modèles mathématiques, les données du problème et le modèle qui s'expriment de la manière suivante :

n : le nombre de tâches.

m : le nombre de machines.

j : l'indice de la tâche, où $j = 1, ..., n$

k : l'indice de la machine, où $k = 1, ..., m$

r : la position de la tâche dans une machine, où $r = 1, ..., nk$

r_{jk} : date de début.

d_{jk} : date de n la tâche j

s_{jk} : temps de préparation de la tâche j effectuée immédiatement après la tâche i sur une machine.

p_{jk} : temps opératoire de la tâche j.

c_{jk} : date de fin d'exécution de la tâche j.

T_{jk} : retard réel de la tâche j.

c max : makespan.

n_k : nombre des tâches affectées à la machine k.

$$Minimiser(c \quad max \quad \sum T) \tag{3.1}$$

sous contraintes :

$$\sum_{k=1}^{n} X_{jkr} \qquad k = 1, 2, ..., m \quad r = 1, 2, ..., n_k \tag{3.2}$$

$$\sum_{j=1}^{m} \sum_{r=1}^{nk} X_{jkr} \qquad j = 1, 2, ..., n \tag{3.3}$$

$$p_{[kr]} = \sum_{j=1}^{n} X_{jkr} p_{jk} r_{ij} \qquad k = 1, 2, ..., m \quad r = 1, 2, ..., n_k \tag{3.4}$$

$$s_{[kr]} = \sum_{i=1}^{n}\sum_{j=1}^{n} X_{jkr} y_{ij} s_{ij} \qquad k = 1, 2, ..., m \quad r = 1, 2, ..., n_k \tag{3.5}$$

$$r_{[kr]} = \sum_{j=1}^{n} X_{jkr} r_{jk} \qquad k = 1, 2, ..., m \quad r = 1, 2, ..., n_k \tag{3.6}$$

$$d_{[kr]} = \sum_{j=1}^{n} X_{jkr} r_{jk} \qquad k = 1, 2, ..., m \quad r = 1, 2, ..., n_k \tag{3.7}$$

$$c_{[kr]} = \max(c_{[k,r-1]} + s_{[kr]}, r_{[kr]}) + p_{[kr]} \qquad k = 1, 2, ..., m \quad r = 1, 2, ..., n_k \tag{3.8}$$

$$t_{[kr]} = \max(c_{[kr]} - d_{[kr]}, 0) \qquad k = 1, 2, ..., m \quad r = 1, 2, ..., n_k \tag{3.9}$$

$$c_{max} = k = 1mr = 1n_k c_{kr} \qquad k = 1, 2, ..., m \quad r = 1, 2, ..., n_k \tag{3.10}$$

$$\sum T_{jk} = \sum_{k=1}^{n}\sum_{r=1}^{n_k} T_{kr} \qquad k = 1, 2, ..., m \quad r = 1, 2, ..., n_k \tag{3.11}$$

$$X_{[jkr]} = 0 \quad or \quad 1 \qquad k = 1, 2, ..., m \quad r = 1, 2, ..., n_k \quad j = 1, 2, ..., n \tag{3.12}$$

$$Y_{[jkr]} = 0 \quad or \quad 1 \qquad k = 1, 2, ..., m \quad r = 1, 2, ..., n_k \quad j = 1, 2, ..., n \tag{3.13}$$

La fonction (1) exprime directement les objectifs de notre problème, i.e, la minimi-sation du makespan et la minimisation de la somme des retards.

Les contraintes (2) et (3) sont des contraintes de singularité des tâches sur la machine k et à la position r. Elles garantissent qu'il y a seulement une tâche sur la machine k et à la position r, et que chaque tâche est déplacée seulement une fois sur ces machines.

La contrainte (4) calcule la durée d'opération de la tâche qui est en position r sur la machine k.

La contrainte (5) définit le temps de préparation de la tâche qui est en position r sur la machine k.

Les contraintes (6) - (9) concernent respectivement la date de début au plus tôt, la date de fin au plus tard, la date de fin d'exécution et le retard réel de la tâche qui est en position r sur la machine k.

Les contraintes (10) et (11) représentent le calcul du makespan et de la somme des retards. La variable binaire X_{jkr} est égale à 1, si la tâche j est ordonnée en position r sur la machine k et 0 sinon. La variable binaire Yij contrôle la position relative de deux tâches i et j. Si la tâche i précède immédiatement la tâche j, alors Y_{ij} est égale à 1, sinon elle est égale à 0. Par contre, si j rst la premier tâche ($j = 1$), alors Y_{ij} est égale à 1, car aucune tâche ne précède j.

Représentation des problèmes d'ordonnancement à machines parallèles :.

Dans les problèmes d'ordonnancement à machines parallèles, on peut afficher les ordonnancements pour nous problèmes avec le diagramme de Gantt, car il est permis de visualiser les séquences des opérations des tâches, en représentant chaque tâche par une ligne sur laquelle sont visibles, les périodes d'exécution des opération et les périodes ou la tâche est en attente des ressources.

Insertion d'une tâche venant aléatoirement :

Une approche basée sur le contexte aléatoire, c'est la méthode d'insertion des tâches dans un ordonnancement prévisionnel. Cette méthode consiste à sélectionner et choisir le bon endroit (emplacement) où ces nouvelles tâches doivent être inclues. Dans la littérature on trouve que l'utilisation de la méthode d'insertion de tâches n'été pas limitée au problème d'ordonnancement d'une seule ressource, mais aussi pour plusieurs ressources (comme les ressources humaines dans les problèmes de d'ordonnancement de la maintenance).

Exemple 3.1. *si l'une des étapes d'assemblage nécessite beaucoup de temps, il serait très intéres- sant alors d'avoir plusieurs machines parallèles qui effectuent la même tâche. D'un autre côté,les machines parallèles sont classées suivant leur rapidité.Si toutes les machines de l'ensemble ont la même vitesse de traitement et effectuent les mêmes tâches, elles sont identiques. Si les machines ont des vitesses de traitement différentes mais linéaires alors elles sont dites uniformes. Dans le cas où les vitesses des machines sont indépendantes les unes des autres, on parle alors de modèle de machines parallèles non reliées ou indépendantes.*

3.2 Les problèmes d'ordonnancement d'atelier

La classification des problèmes d'ordonnancement dans un atelier peut se fait selon le nombre de machines et leur ordre d'utilisation pour fabriquer un produit, qui dépend de la nature de l'atelier considéré. Un atelier est caractérisé par le nombre de machines qu'il contient et par son type.

Dans ce chapitre, nous présenterons la définition du problème de l'atelier, quelques concepts sur les types d'ateliers, et présenterons également les relations entre ces types et à la fin nous pouvons voir la complexité et les types de problèmes.

Définition 3.2. *Dans un problème d'atelier, une pièce doit être usinée ou assemblée sur différentes machines. Chaque machine est une ressource disjonctive, c'est-à-dire qu'elle ne peut exécuter qu'une tâche à la fois, et les tâches sont liées exclusivement par des contraintes d'enchaînement.*

Plus précisément, les tâches sont regroupées en « n » entités appelées travaux ou lots. Chaque lot est constitué de « m » tâches à exécuter sur « m » machines distinctes, et dans le cas des problèmes d'atelier, une tâche est une opération, une ressource est une machine et chaque opération nécessite pour sa réalisation une machine.

Dans le modèle de base de l'ordonnancement d'atelier, l'atelier est constitué de « m » machines, « n » travaux (jobs) , disponibles à la date 0, doivent être réalisés, un travail i est constitué de ni opérations, l'opération j du travail i est notée (i, j) *avec* $(i, 1) < (i, 2) < ... < (i, ni)$ *si le travail i possède une gamme (* $A < B$ *signifie A précède B). Une opération*

(i, j) *utilise la machine* m_{ij} *pendant toute sa durée* p_{ij} *et ne peut être interrompue .*

Un atelier se définit par le nombre de machines qu'il contient et par son type. Une classification peut exister selon le nombre des machines et l'ordre d'utilisation des machines, pour réaliser un travail (par exemple fabrication d'un produit qui dépend de la nature de l'atelier).[6].

Solon les caractéristiques des machines on peut distinguer plusieurs types des problèmes :

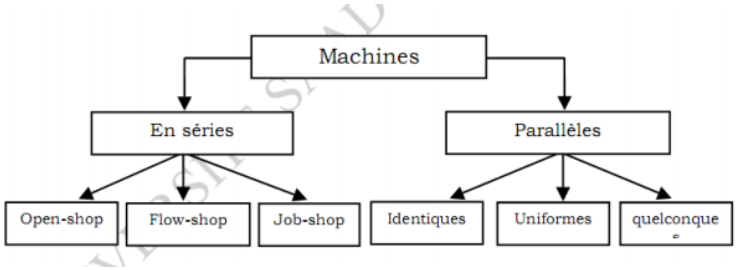

FIGURE 3.1 – Les types des machines

3.3 Les types des Problèmes

Les problèmes d'ordonnancement sont généralement classés en deux principaux modèles dépendamment du nombre d'opérations que requièrent les jobs : des modèles à une opération (machine unique et machines parallèles) et des modèles à plusieurs opérations (flow-shop, open shop et job shop).

3.3.1 Modèles à une opération

Modèle à machine unique

Dans un modèle à machine unique, l'ensemble des tâches à réaliser est exécuté par une seule machine. L'une des situations intéressantes où on peut rencontrer ce genre de configurations est le cas où on est devant un système de production comprenant une machine goulot qui influence l'ensemble du processus. .[7].

Ce modèle est illustré dans la Figure 3.2.

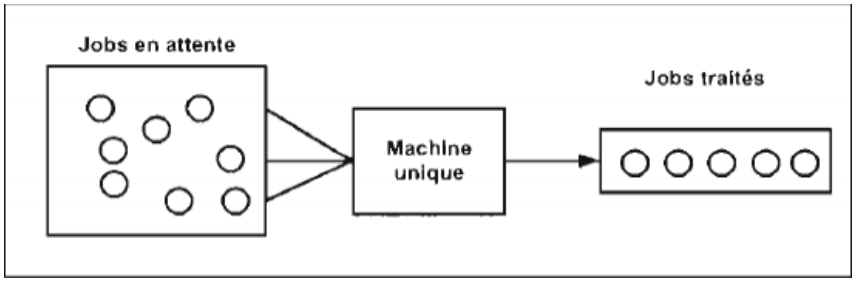

FIGURE 3.2 – Modèle à machine unique.

Modèle à machines parallèle

Le deuxième modèle est le modèle à machine parallèle. Ce modèle est utilisé surtout dans les secteurs industriels tels que : l'industrie alimentaire, les industries plastiques, les fonderies et en particulier l'industrie textile. Le processus de déroulement de ce système de production est le suivant : à chaque fois qu'une machine i se libère, ont lui affecte un job j comme illustré à la Figure 3.3.

Dans le cas d'un processus d'assemblage industriel, par exemple, si l'une des étapes d'assemblage nécessite beaucoup de temps, il serait très intéressant alors d'avoir plusieurs machines parallèles qui effectuent la même tâche.

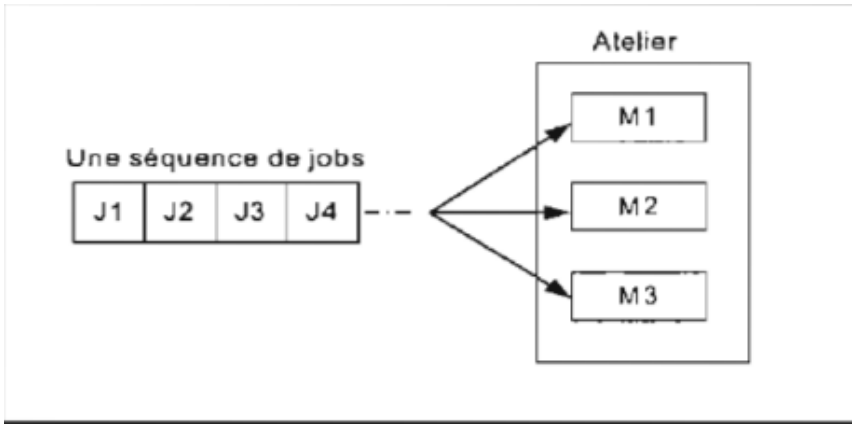

FIGURE 3.3 – Modèle à machines parallèle.

*Figure*3.3 : *Modèle* à *machines parallèle.*

Ce type d'atelier peut être divisé en trois sous-catégories selon la vitesse d'exécution des machines :

1. les ateliers à machines identiques : toute tâche peut s'exécuter sur n'importe quelle machine avec une même durée opératoire.
2. les ateliers à machines uniformes : chaque machine possède sa propre vitesse, indépendamment de la durée de la tâche à exécuter.
3. les ateliers à machines indépendantes : la vitesse de chaque machine dépend de l'opération à effectuer.[1].

3.3.2 Modèles à plusieurs opérations

Le modèle à plusieurs opérations est constitué des cas où un job, pour se réaliser, doit passer par plusieurs machines, chacune de ces machines ayant ses spécificités. On distingue trois modèles selon l'ordre de passage des jobs sur les machines, à savoir les modèles de flow-shop, job-shop et open-shop.

Modèle flow-shop

Dans le modèle de flow-shop, les ordres de fabrication visitent les machines dans le même ordre, avec des durées opératoires pouvant être différentes. Chaque job va être s'exécuter sur les M machines en série et tous les jobs vont suivre le même ordre de passage sur ces machines. Ce type de modèle est aussi appelé modèle linéaire.

La figure 3.4 illustre le cas d'un flow-shop avec quatre machines et quatre jobs. Les quatre jobs suivent le même ordre de traitement sur les quatre machines.[7].

3.3.3 Modèle job-shop

Concernant le modèle de job-shop, chaque job à un ordre à suivre et chacun d'entre eux peut s'exécuter plusieurs fois sur la même machine ; ce qui n'est pas le cas du flow-

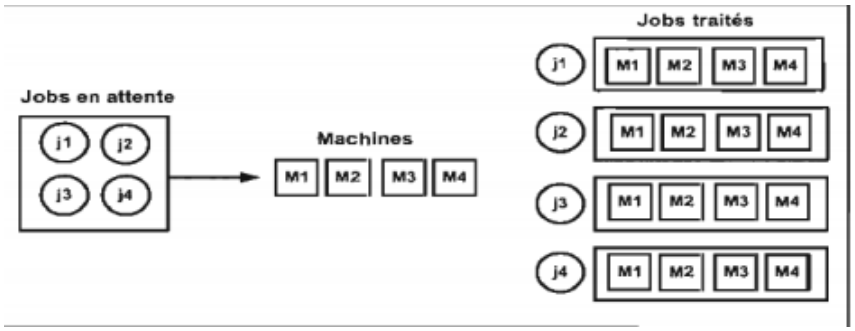

FIGURE 3.4 – Modèle flow-shop.

shop. Il s'agit dans ce cas de déterminer les dates de passage sur différentes ressources d'ordres de fabrication ayant des trajets différents dans l'atelier.

Ces ordres de fabrication partageant des ressources communes, des conflits sont susceptibles de survenir, résultant des croisements de flux illustrés à la Figure 3.5.

Dans cette figure, l'ordre de passage de chaque job est indiqué par un chemin de même couleur que celui du job. Par exemple, le job J1 passe par toutes les machines, alors que le job J3 ne passe que par la deuxième et la quatrième machine. Dans son expression la plus simple, le problème consiste à gérer ces conflits tout en respectant les contraintes données, et en optimisant les objectifs poursuivis.

Les types de ressources et de contraintes prises en compte peuvent toutefois considérablement compliquer le problème. Plus on intégrera de contraintes, plus on se rapprochera d'un cas réel, mais moins on disposera de méthodes de résolution satisfaisantes.[7].

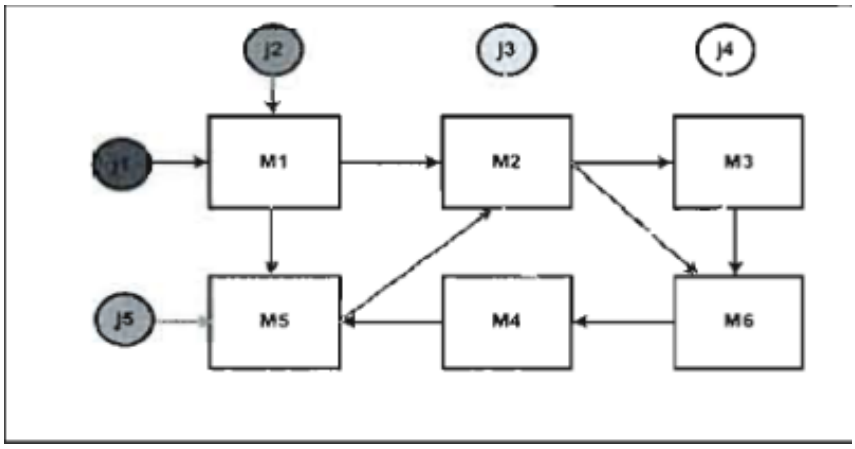

FIGURE 3.5 – Modèle job-shop.

Modèle open-shop

Dans le modèle d'open shop, l'ordre de passage des n jobs sur les m machines n'est pas connu à l'avance. Cet ordre est déterminé lors de la construction de la solution. Chaque job j peut avoir son propre ordre de passage sur toutes les machines.

Le fait qu'il n'y ait pas d'ordre prédéterminé rend la résolution du problème d'ordonnancement de ce type plus complexe, mais offre cependant des degrés de liberté intéressants.

À la Figure 3.6, nous avons un ensemble de quatre jobs et un ensemble de quatre machines. À droite de la figure nous pouvons remarquer que chaque job a suivi un ordre de passage différent sur les quatre machines.

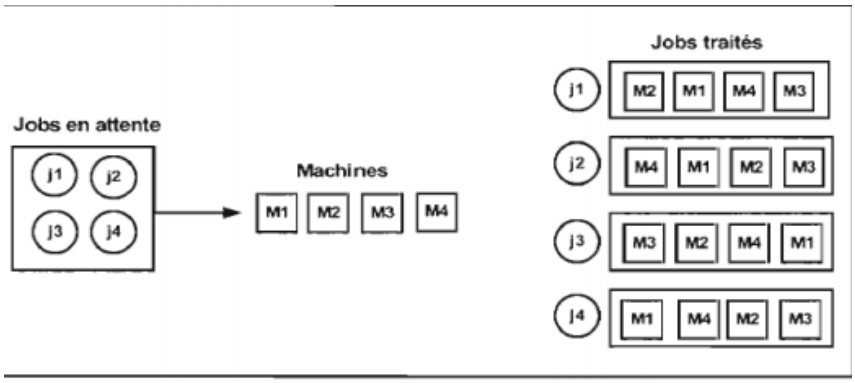

FIGURE 3.6 – Modèle open-shop.

Conclusion Générale

Dans ce mémoire,d'ordonnancement, tout d'abord, intéressés à l'étude des problèmes de l'optimisation combinatoire avec leurs méthodes de résolution pour trouver de bonnes solutions aux problèmes correspondants, à savoir des méthodes exactes (programmation linéaire, Branch Bound, programmation dynamique) et des méthodes approchées (recuit simulé, recherche tabou, algorithmes génétiques). Nous sommes penchés sur les problèmes d'ordonnancement dans sa globalité, avec ses différentes composantes (tâches, ressources, contraintes, critères), et aussi avec les différents types d'ateliers étudiés.

Bibliographie

[1] N . El Hage Hassan , *Topologie générale et espaces normés* , Dunod , Paris , 2011 .

[2] G . Aissa cours d'lgébre premier année licence , Universite Mohamed Boudiafde de M'sila , 2016 : 2017

[3] A.Karray. « Contribution à l'ordonnancement d'ateliers agroalimentaires utilisant des méthodes d'optimisation hybrides , thèse de doctorat ,Ecole centrale de Lille Université de Tunis El Mmanar Ecole Nationale D'ingénieurs de TUNIS ». (2012)

[4] B.Merouane Hocine.«les Problemes d' Ordonnancement a Machines Paralleles, de Taches Dependantes, Universite Saad Dahlab de Blida».(2006)

[5] B.M. E., Optimisation à base de simulation pour le développement des syst-mes décisionnels, Thèse Doctorat 3éme cycle en informatique, université de M'sila.(2015)

[6] B.MOHAMMED, «Problèmes d'Ordonnancement d'Atelier, Memoire de fin d'etude, université Sidi Mohamed Ben Abdellah » .(2016)

[7] A.Gherboudj « Méthodes de résolution de problèmes difficiles académiques, Thèse du diplôme de Doctorat, Université de Constantine2». (2013) .

[8] C. Abderrahman et Dahmani Abd Naceur«Ordonnancement d'un flow-shop par métaheuristique hybride , Mémoire de Fin d'Etudes , Université Abou Bekr Belkaid Tlemcen » . (2017)

[9] G.Abderrahim «Application d'une approche BIO-INSPIREE au Problème d' Ordonnancement des instructions, diplôme de Magistère, Université de M'sila,».(2012)

[10] Hamdy A. T. « Operations Research : An Introduction, 8th ed., Prentice Hall ». (2007)

[11] H.Aboubakr et LEMMOUIS Abdelhamid«Résolution d'un problème d'ordonnancement de type job shop avec contrainte de transport, Projet de Fin d'Etudes, Université Abou Bekr Belkaid Tlemcen ».

[12] I.LARIBI «Résolution de problèmes d'ordonnancement de type Flow-Shop de permutation en présence de contraintes de ressources non-renouvelables, Présentée pour l'obtention du grade de DOCTEUR , Université Tlemcen » .(2018)

[13] J.W. Herrman, et al. Handbaook of Production Scheduling, Springer NY.(2006.)

[14] K.Mebarek «Utilisation des stratégies Métaheuristiques pour l'ordonnancement des ateliers de type Job Shop, Mémoire de Magister, Université El-Hadj Lakhdhar atna» .(2008)

[15] M.C. Cooper . «Théorie de la complexité IRIT, Université de Toulouse 3» .

[16] M.Meziane E. A., Optimisation par phases pour les problèmes d'ordonnancement des ateliers de type job-shop totalement flexibles, Mémoire de magister en des sciences, Université d'Oran, Algérie .(2011)

[17] S.Ourari, « L'ordonnancement déterministe à l'ordonnancement , Thèse en vue de l'obtention du garde de docteur de l'Université de TOULOUSE».(2011)

[18] S. Mehdi« Investigations sur la sélection de routages alternatifs en temps réel basées sur les méta-heuristiques-les essaims particulaires, Thèse de Doctorat en Sciences en Productique, Université de Tlemcen». (2012)

[19] T. Vallée, M. Yıldızoglu «Présentation des algorithmes génétiques et de leurs applications en économie, Université de Nantes, LEA-CIL».

[20] W.LABBI et Mourad BOUDHAR « Méta heuristiques pour un problème d'ordonnancement sous contraintes de préparation Laboratoire RECITS, Faculté de Mathématiques, USTHB BP 32 El-Alia, BEZ, Alger, Algérie» .(2015) *Les sites web* .

[1] https://www.techno-science.net/glossaire-definition/Complexite.html#ref_1

[2] https://fr.wikipedia.org/wiki/Algorithme_glouton

Abstract

In this mémoire, we study the issue of scheduling on a group of parallel machines At first, we reviewed a set of definitions, then the well-known solutions, and finally we gave some examples about scheduling models.

Key words : Scheduling, Parallel machines, Methode of solution.

ملخص

في هذه المذكرة قمنا بدراسة مسألة الجدولة على مجموعة من الالات المتوازيةيداية استعرضنا مجموعة من التعريفات ثم طرق الحل المعروفة واخيرا قمنا باعطاء بعض الأمثلة حولنماذج الجدولة

الكلمات المفتاحية: جدولة ، الات متوازية ، طرق الحل