الجمهورية الجزائرية الديمقراطية الشعبية

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

وزارة التعليم العالي و البحث العلمي

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

جامعة أبي بكر بلقايد– تلمسان

Université Aboubakr Belkaïd-Tlemcen

كلية التكنلوجيا

Faculté de Technologie

Département de Génie Electrique et Electronique (GEE)

Filière : Génie Industriel

MASTER EN GENIE INDUSTRIEL

Option ingénierie de production

PROJET DE FIN D'ETUDES

Présenté par:

Sahel Faiçal

Intitulé du Sujet

Etude de performance des règles de priorité dans la résolution de problème d'ordonnancement job shop

Soutenu devant le jury composé de :

Houbad Yamina	**MAA**	Univ. Tlemcen	Présidente
Belkaid Fayçal	**MCA**	Univ.Tlemcen	Examinateur
Sekkal djazia	**Doctorante**	Univ.Tlemcen	Examinatrice
Hadri Abdelkader	**MAA**	Univ. Tlemcen	Encadreur

Année Universitaire: 2020/2021


بسم الله الرحمن الرحيم

Dédicace

A ma très chère Mère

Affable, honorable, aimable : Tu représentes pour moi le symbole de la bonté par excellence, la source de tendresse et l'exemple du dévouement qui n'a pas cessé de m'encourager et de prier pour moi. Ta prière et ta bénédiction m'ont été d'un grand secours pour mener à bien mes études.

A mon Père

Aucune dédicace ne saurait exprimer l'amour, l'estime, le dévouement et le respect que j'ai toujours eu pour vous. Rien au monde ne vaut les efforts fournis jour et nuit pour mon éducation et mon bien être.

A ma petite sœur Meriem

A mes frères Ramzi & Oussama

A mes chèrs ami (e)s.

Remerciement

Je remercie ALLAH le tout puissant de m'avoir donné le courage, la volonté et la patience de mener à terme le présent travail.

Avant de commencer la présentation de ce travail, je profite, l'occasion pour remercier toutes les personnes qui ont contribué de près ou de loin à la réalisation de ce projet de fin d'études.

Je tiens à exprimer mes vifs remerciements pour mon respectueux encadreur monsieur Hadri Abdelkader. Vous avez bien voulu me confier ce travail riche d'intérêt et me guider à chaque étape de sa réalisation.

Je voudrais également remercier les membres des jurys d'avoir accepté d'évaluer ce travail et pour toutes leurs remarques et critiques, Madame Houbad Yamina Maitre assistante à l'Université de Tlemcen, qui nous a fait l'honneur de présider notre jury de mémoire, et Monsieur Belkaid fayçal, Maitre de Conférences à l'Université de Tlemcen, et madame Sekkal Djazia doctorante à l'université de Tlemcen d'avoir accepté d'examiner notre travail.

Mes remerciements vont aussi à tous mes professeurs, enseignants et toutes les personnes qui m'ont soutenus jusqu'au bout.

Table des matières

Liste des tableaux

Liste des figures

Introduction générale

En industrie, la fabrication d'un produit doit passer par plusieurs opérations sur des machines. Donc, pour un ensemble de produits on est confronté à un problème de choix de l'ordre de passage, la question qui se pose, Dans quel ordre doivent passer les produits ? Ceci revient à un problème d'ordonnancement.

L'ordonnancement sert à programmer l'exécution des tâches (opérations) sur un ensemble de ressources (humaines et machines) à fin de déterminer un plan optimal d'implantation, et optimiser certain critères de temps, budget...etc.

Pour un atelier job shop le croissement de nombre de jobs et de machines sert à le croissement de difficulté de problèmes d'ordonnancement, où les méthodes exactes redeviennent obsolètes pour résoudre ce genre de problèmes. Ce genre de problèmes appelée les problèmes NP-difficiles, donc Elles nécessitent l'utilisation des heuristiques pour fournir des solutions assez proches que possible de la solution optimale.

Dans notre travail nous intéressons à la résolution d'un problème d'ordonnancement de type job shop avec minimisation de makespan par l'utilisation des heuristiques, nous allons appliquer les heuristiques SPT et LPT système qui donne un séquencement des jobs pour tous le système et l'heuristique shifting bottleneck, SPT et LPT sur les machines qui définissent un ordre des jobs pour chaque machine de notre atelier job shop. L'application de ces méthodes sera effectuée sur des problèmes différents tailles. Ce document s'articule autour de trois chapitres :

Dans le premier chapitre de ce mémoire on parle des généralités d'ordonnancement, la notion de complexité de problèmes combinatoires et les types d'ateliers pour un système de production.

Dans le deuxième chapitre nous intéressons aux méthodes d'optimisation, on parle de manière générale de l'atelier job shop.

Dans le troisième chapitre nous présentons le système et le problème étudié, les heuristiques utilisées pour résoudre ce type de problème puis nous montrons la simulation et les résultats obtenus après l'application de ces méthodes. Ensuite, nous présentons la discussion et nos commentaires sur les résultats. Enfin, on termine par une conclusion générale de notre travail.

I Chapitre 1 : Généralités sur l'ordonnancement

I.1 Introduction

Dans le domaine industriel, l'ordonnancement consiste à trouver une solution optimal par le planning d'exécution des taches en affectant simultanément les ressources nécessaires pour réaliser ces taches au cours de temps et en respectant certains contraintes de production (retard, priorités, contraintes d'enchaînement).

Dans ce chapitre, nous présentons les notions fondamentales concernant les systèmes de production, les notions liées à l'ordonnancement, à savoir la définition d'un problème d'ordonnancement, les objectifs de l'ordonnancement et les types d'ateliers ainsi que on va voire la notion de complexité de problème d'optimisation.

I.2 Les systèmes de production

Un system de production est l'ensemble des moyens et des ressources utilisés pour fabriquer des produits finis à partir de produits bruts en produits de valeur supérieure qui peuvent être [1]:

-Des produits finis, directement commercialisés.
-Des produis intermédiaires, servant à la réalisation de produits fini.

Plus précisément, la charge d'un system de production est constituée de jobs. Un job suit la réalisation d'un produit dans toutes les étapes de production (de la matière première jusqu'au produit fini). Chaque job a une gamme qui décrit la suite des opérations qu'il doit réaliser. La gamme décrit les opérations à réaliser en donnant [1]:

-Leur durée,
-La ou les machines qui doivent réaliser cette opération,
-Les ressources consommées (type de ressources et quantité utilisée),
-L'ordre entre les opérations à réaliser (facultatif).

I.3 Généralités sur l'ordonnancement

Ordonnancer le fonctionnement d'un système industriel de production consiste à gérer l'allocation des ressources au cours du temps, tout en optimisant au mieux un ensemble de

critères. C'est aussi programmer l'exécution d'une réalisation en attribuant des ressources aux tâches et en fixant leurs dates d'exécution. [2] [3]

Ordonnancer peut également consister à programmer l'exécution des opérations en leur allouant les ressources requises et en fixant leurs dates de début de fabrication.

L'ordonnancement se déroule en trois étapes qui sont :

-La planification : qui vise à déterminer les différentes opérations à réaliser, les dates correspondantes, et les moyens matériels et humains à y affecter.
-L'exécution, qui consiste à mettre en œuvre les différentes opérations définies dans la phase de planification.
-Le contrôle, qui consiste à effectuer une comparaison entre planification et exécution, soit au niveau des coûts, soit au niveau des dates de réalisation.

I.4 Les éléments d'un problème d'ordonnancement

La modélisation d'un problème d'ordonnancement passe par la spécification des tâches et leurs caractéristiques, des ressources et des critères à optimiser. Nous allons dans ce qui suit définir les quatre notions fondamentales qui interviennent dans un problème d'ordonnancement : les tâches (Travaux ou Jobs en anglais), les ressources, les objectifs et les contraintes.

I.4.1 Les tâches

Une tâche i est une entité élémentaire de travail localisée dans le temps par une date de début S_i et/ ou de fin C_i, dont la réalisation nécessite une durée $P_I = C_i - S_i$, et qui consomme des moyens [4].On distingue deux types de tâches :

- les tâches morcelables (préemptives) qui peuvent être exécutées en plusieurs fois, facilitant ainsi la résolution de certains problèmes [4].
- les tâches non morcelables (indivisibles) qui doivent être exécutées en une seule fois et ne sont interrompues qu'une fois terminées [4].

Une tâche $\mathbf{t_{ij}}$ est localisée dans le temps par :

$\mathbf{p_{ij}}$: La durée opératoire de la tâche i sur la machine j;
$\mathbf{r_{ij}}$: La date de début au plutôt ou date de disponibilité de la tâche i sur la machine j ;

d_{ij}: La date de fin au plus tard ou «deadline» de la tâche i sur la machine j;

t_{ij} : La date de début d'exécution de la tâche i sur la machine j ;

c_{ij} : La date de fin d'exécution de la tâche i sur la machine j ;

T_{ij} : Le retard vrai de la tâche i sur la machine j / T_{ij} = max (0,cij – dij);

U_{ij} : L'indice de retard Tel que U_{ij} =1 si la tâche est en retard, 0 sinon ;

$g_{ij}(u)$: Le coût attaché à la fin d'exécution de la tâche i sur la machine j, à la date u.

La figure I illustre les caractéristiques temporelles présentées ci-dessus :

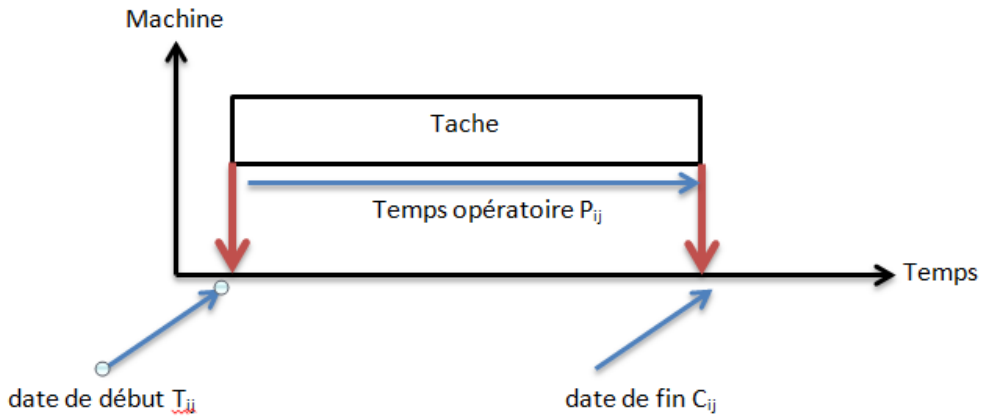


Figure 1: Caractéristiques d'une tâche faisant référence à l'exécution d'une opération.

I.4.2 Les ressources

Une ressource est un moyen technique ou humain, destiné à être utilisé pour la réalisation d'une tâche et disponible en quantité limitée. On note en général M= {M_1, M_2,…, M_m} l'ensemble des ressources. Plusieurs types de ressources sont à distinguer [6].

Une décomposition de type ressource est illustré par la figure 2.

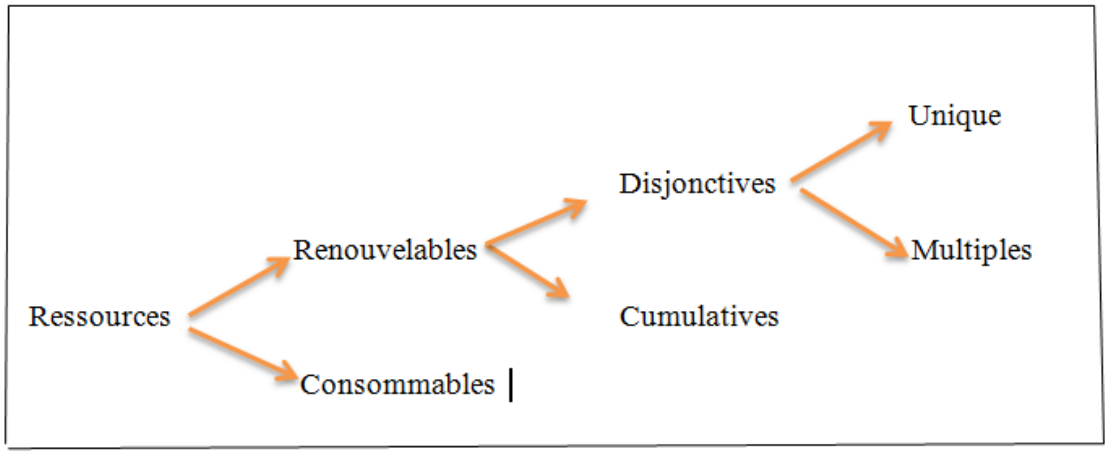


Figure 2: Décomposition des ressources.

I.4.2.1 Les ressources renouvelables

Une ressource renouvelable est une ressource qui est utilisée par une tache et reste disponible après la fin de cette dernière en même quantité pour l'utilisation par les taches restantes, par exemple de ressources renouvelable : les machines, les employés,…

-Ressources disjonctives :

Sont des ressources non partageables, elles exécutent par une seule tâche à la fois,

-Ressources cumulatives :

Sont des ressources partageables, elles exécutent par plusieurs tâches simultanément.

I.4.2.2 Les ressources non renouvelables

Une ressource est non renouvelable (ou consommable), est une ressource consommée complètement au cours de temps par une certaine tache donc ne sera pas disponible à l'utilisation par les autres taches restantes, par exemple : la matière première est une ressource consommable.

I.4.3 Les contraintes

Une contrainte exprime des restrictions sur les valeurs que peuvent prendre conjointement les variables représentant les relations qui relient les tâches et les ressources. On distingue les contraintes temporelles et les contraintes de ressources. [5]

Les contraintes temporelles intègrent:

-Les contraintes de temps alloué, issues généralement d'impératifs de gestion et relatives aux dates limites des tâches (délais de livraison, disponibilité des approvisionnements) ou à la durée totale d'un projet.

-Les contraintes d'antériorité et plus généralement les contraintes de cohérence technologique, qui décrivent le positionnement relatif de certaines tâches par rapport à d'autres.

-Les contraintes de calendrier liées au respect d'horaires de travail, etc.

I.4.4 Les critères

Lors de la résolution d'un problème d'ordonnancement, on peut distinguer entre deux types de stratégies, visant respectivement à l'optimalité des solutions par rapport à un ou plusieurs critères, ou à leur admissibilité vis-à-vis des contraintes. L'approche par optimisation suppose que les solutions candidates à un problème puissent être ordonnées de manière rationnelle selon un ou plusieurs critères d'évaluation numérique permettant d'apprécier la qualité des solutions. [5]

Quelques exemples sont cités ci-dessous : [3]

-La minimisation des dates d'achèvement des actions.

-La minimisation du maximum des dates d'achèvement des actions.

-La minimisation de la moyenne des dates d'achèvement des actions.

-La minimisation des retards sur les dates d'achèvement des actions.

-La minimisation du maximum des retards sur les dates d'achèvement des actions.

-La minimisation des encours.

-La minimisation du coût de stockage des matières premières.

-L'équilibrage des charges des machines.

-L'optimisation des changements d'outils.

La satisfaction de tous les critères à la fois est souvent délicate, car elle conduit souvent à des situations contradictoires et à la recherche de solutions à des problèmes complexes d'optimisation. [7]

I.5 Formulation de problème d'ordonnancement

La formulation de problème d'ordonnancement en trois champs : $\alpha \mid \beta \mid \gamma$

- Le champ α spécifie l'environnement machines et contient une seule entrée (1, Pm, Fm, Om, Jm…)
- Le champ β spécifie les caractéristiques et contraintes d'opérations. Il peut contenir (r_j, prmp, prmu…)
- Le champ γ spécifie l'objectif à atteindre, exemple: (C_{max}, L_{max}…)

Le champ α : environnement machines

- Machine unique (1)
- Machines en parallèle (Pm, Qm, Rm)
- Flow shop (Fm)
- Flow shop flexible (FFs)
- Job shop (Jm)
- Open shop (Om)

<<m>> représente le nombre de machines.

Le champ β : caractéristiques et contraintes

- Dates de disponibilité (r_j) : disponibilité des taches.
- Préemption (prmp) : les opérations peuvent être exécutées par morceaux.
- Précédence (prec) : certaines taches doivent être exécutées avant d'autres.
- Pannes (brkdwn) : les ressources (machines) ne sont pas disponibles en permanence.
- Permutation (prmu) : il n'y a pas de permutation possible entre les taches dans un flow shop.

Le champ γ : fonction objectif

- Makespan (C_{max}) : date de fin de la dernière tache
- Maximum lateness (Retard algébrique) (L_{max}) :

$$\text{Max } (C_j - d_j)$$

- Total weighted completion time ($\sum w_j C_j$) :
- Total weighted tardiness ($\sum w_j T_j$) :

$$T_j = \max (C_j - d_j, 0)$$

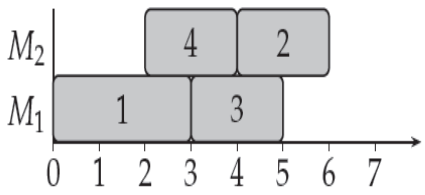
- Weighted number of tardy jobs ($\sum w_j U_j$) :

$U_j=1$ si $C_j>d_j$ et 0 si $C_j<=d_j$

I.5.1 Classification des ordonnancements

I.5.1.1 Ordonnancement semi-actif

Lorsqu'aucune opération ne peut être terminée plus tôt sans affecter les séquences d'opération de l'une des machines.

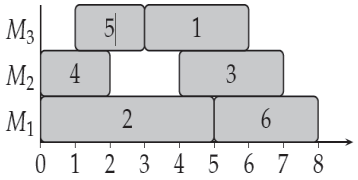


job *i*	1	2	3	4
r_i	0	0	1	2
p_i	3	2	2	2

Figure 3: diagramme de gantt pour exemple d'ordonnancement semi-actif.

I.5.1.2 Ordonnancement actif

Lorsqu'aucune opération ne peut être terminée plus tôt sans retarder une autre opération.



job i	1	2	3	4	5	6
r_i	0	0	4	0	1	2
p_i	3	5	3	2	2	3

Figure 4: diagramme de gantt pour exemple d'ordonnancement actif.

I.5.1.3 Ordonnancement sans délais

Aucune machine ne doit rester libre pendant qu'une opération est en attente.

- Pour la majorité des problèmes, il existe des ordonnancements optimaux sans délais.
- Tous les problèmes avec préemption les ordonnancements optimaux sont sans délais.
- Dans certains problèmes non préemptifs il est intéressant de provoquer des arrêts forcés des ressources

I.6 Les ateliers de production

Dans les systèmes industriels nous trouvons différentes types d'ateliers de productions, où chaque atelier se caractérise par le nombre et le type d'implantation des machines et l'ordre de passage des produits à fabriquer.

I.6.1 L'atelier à une machine

Consiste à ordonnancer, sur une seule machine, des jobs constitués d'une seule opération pour la minimisation du makespan. L'atelier est représenté par la figure 5.

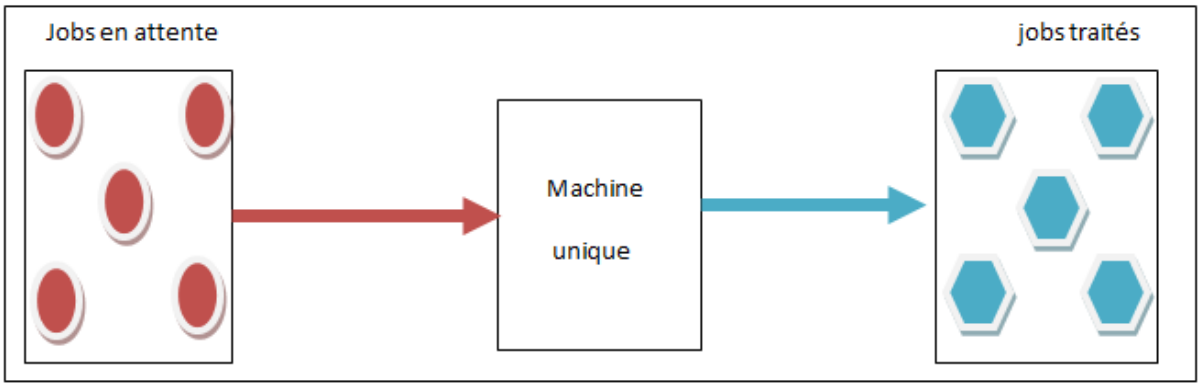


Figure 5: Ordonnancement à une machine.

I.6.2 L'atelier à machines parallèles

Est une généralisation du problème d'atelier à une machine et un cas particulier de problème d'atelier multi machines. Chaque job est constitué d'une seule opération et chaque opération peut être réalisée par n'importe laquelle des machines, disposées en parallèle, mais n'en nécessite qu'une seule. Un exemple de l'atelier est illustré par la figure 6.

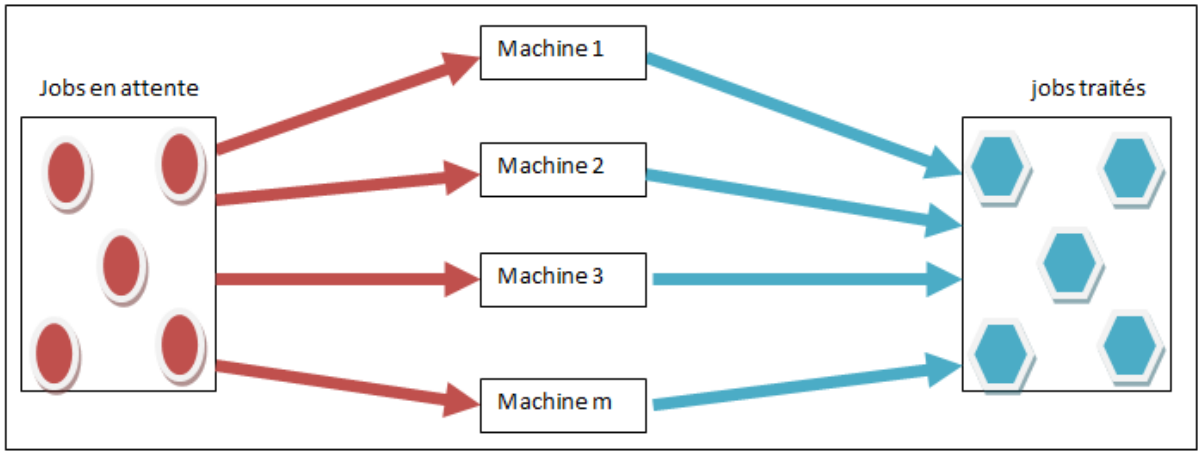


Figure 6: Ordonnancement à machines parallèles.

I.6.3 L'atelier flow-shop

Dans un tel atelier, appelé aussi atelier en ligne ou à cheminement unique, toutes les opérations passent par toutes les machines dans le même et unique ordre. Dans un atelier de type flow-shop : l'ensemble des produits initiaux {Pi} passe successivement sur toutes les machines $M_1 \ldots M_5$ pour donner un ensemble de produits finis {Pf} [8]. La figure 7 représente exemple de l'atelier flow shop.

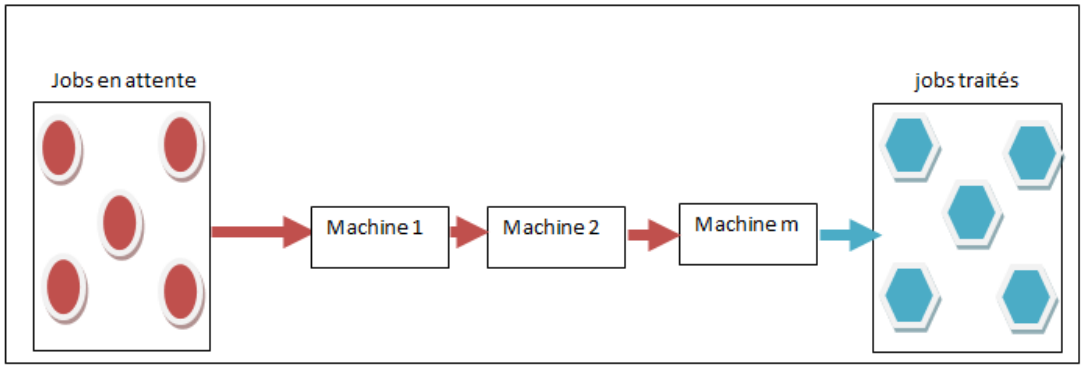


Figure 7: Ordonnancement d'atelier flow shop.

I.6.4 L'atelier job shop

Dans cette classe d'ateliers, appelés aussi ateliers à cheminements multiples, chaque tâche possède son propre mode de passage sur les machines Le Job-shop flexible est une extension du modèle job shop classique. Sa principale particularité réside dans le fait que plusieurs machines sont potentiellement capables de réaliser un sous ensemble d'opérations. Une opération est associée, plus précisément à un ensemble contenant toutes les machines pouvant effectuer cette opération.

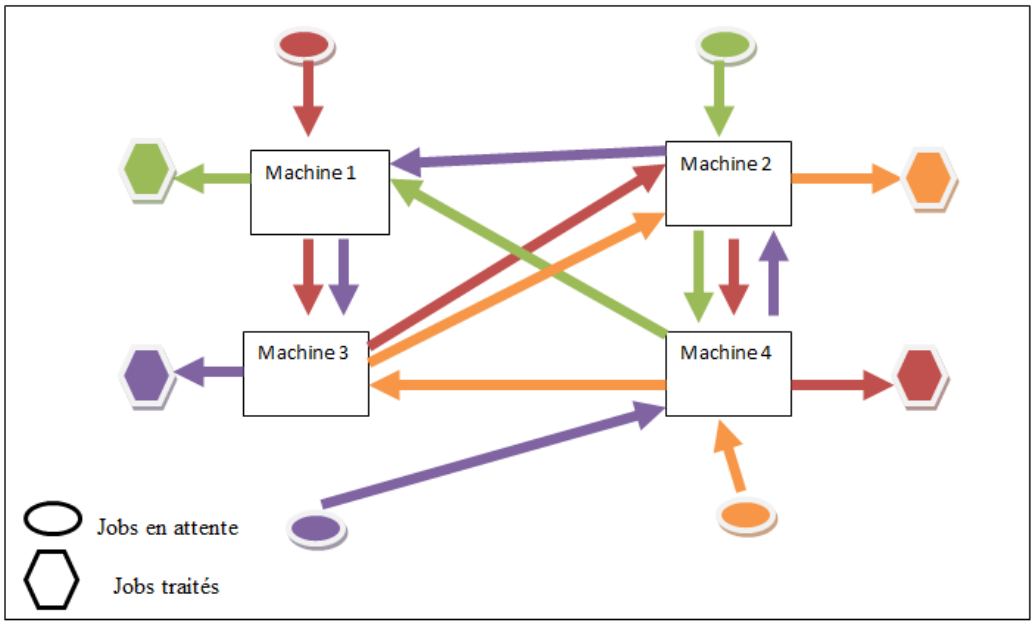


Figure 8: Ordonnancement d'atelier job shop.

I.6.5 L'atelier open-shop

C'est un atelier à cheminement libre, les gammes opératoires des différents jobs ne sont pas fixées a priori. Les opérations d'un même job peuvent donc être exécutées dans un ordre quelconque.

I.7 Notion de complexité de problèmes

La théorie de la complexité a pour but d'apporter des informations sur la difficulté théorique d'un problème à résoudre. Elle permet de classer "du point de vue mathématique" les problèmes selon leur difficulté [9]. La complexité analyse le temps nécessaire pour obtenir une solution.

Pour La résolution d'un problème d'ordonnancement, il faut prouver l'existence d'une solution, ou bien la construire, car la difficulté ce trouve autour la construction de la solution plus que l'improuvement de sa existence et qu'elle nous aide à la classification de problèmes comme étant difficiles ou faciles.

I.7.1 La classe NP

Dans cette classe on trouve les problèmes de décision pour lesquelles il existe des algorithmes polynomiaux non déterministes qu'ils traitent [10].

Cette classe contient deux sous classes : la classe NP-Complet et la classe NP-Difficile

I.7.2 La classe P

Ce sont des problèmes polynomiaux s'il existe un algorithme de complexité polynomiale pour leur résolution.

La classe P est l'ensemble de tous les problèmes de reconnaissance polynomiaux. Pour le reste de la classe NP, on n'est pas sûr qu'il n'existe pas un algorithme polynomial pour résoudre chacun de ses problèmes. Ainsi, on sait que P est inclue dans NP mais on n'a pas pu prouver que P n'est pas NP. [11]

I.7.3 La classe NP-complet

Un problème est NP-complet lorsqu'il peut être résolu par une classe restreinte d'algorithmes de recherche de force brute et qu'il peut être utilisé pour simuler tout autre problème avec un algorithme similaire. ... L'ensemble des problèmes NP-complets est souvent désigné par NP-C ou NPC.

I.7.4 La classe NP-difficile

En informatique théorique, un problème NP-difficile est un problème vers lequel on peut ramener tout problème de la classe NP par une réduction polynomiale.

S'il est également dans la classe NP, on dit que c'est un problème NP-complet.

I.8 Conclusion

Dans ce chapitre nous avons défini le système de production, un problème d'ordonnancement qui est défini par un ensemble de taches à réaliser sur un ensemble de ressources (machines) de sorte qu'une fonction objectif soit optimisée en respectant certaines contraintes. L'ordonnancement peut s'appliquer sur n'importe quelle classe de problème d'optimisation facile ou difficile pour avoir des résultats de bonne qualité.

II Chapitre 2 : les méthodes d'optimisation combinatoires

II.1 Introduction

Les problèmes d'atelier job shop où bien les ateliers à cheminement multiples sont généralement des problèmes NP-difficiles et NP-complet, la résolution de ce genre des problèmes nécessite des méthodes approchées, on parle donc des heuristiques et des metaheuristiques.

Dans ce chapitre, on parle de méthodes d'optimisation combinatoire, ensuite nous intéressons à l'atelier job shop, sa définition, la problématique d'ordonnancement de cet atelier et sa représentation graphique.

II.2 Les méthodes d'optimisation

La majorité des problèmes d'ordonnancement notamment dans le domaine industriel sont des problèmes très compliqués à résoudre, pour résoudre ce type de problèmes et de trouver un ordonnancement acceptable dans un temps raisonnable on utilise des méthodes approchées.

Pour des problèmes de petite taille, nous pouvons obtenir une solution exacte et optimale par l'une des méthodes exactes.

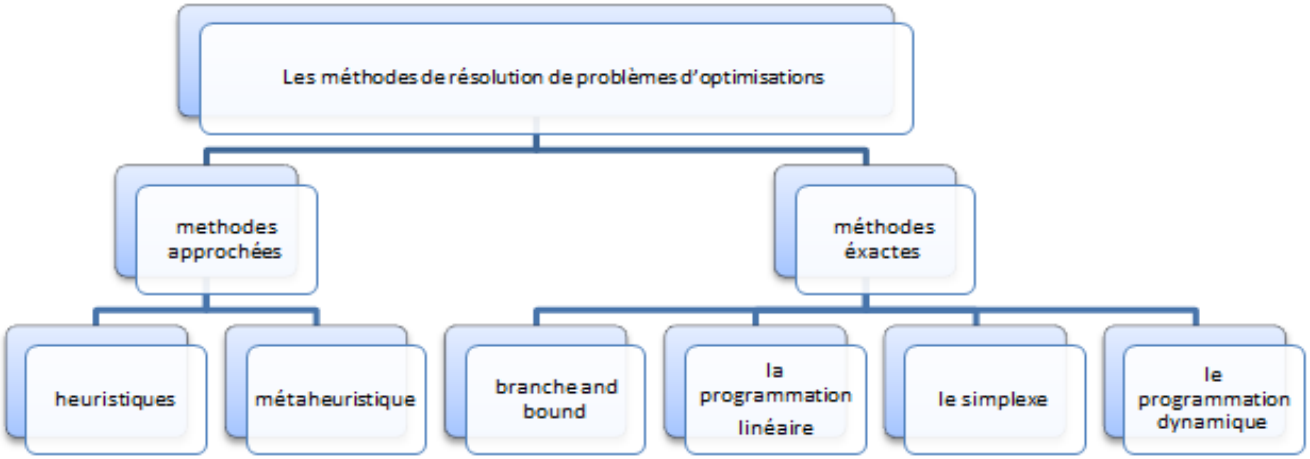


Figure 9: Classification de méthodes de résolution de problèmes d'optimisation.

II.2.1 Les méthodes exactes

Les méthodes exactes de résolution sont des méthodes qui fournissent une solution exacte et optimale en se basant sur des lois bien déterminées [12]. Ces méthodes permettent d'avoir des solutions vis-à-vis un certain critère, sous des conditions bien particulières. Dans ce contexte on trouve par exemple la méthode la plus utilisée pour résoudre les problèmes d'optimisation est la méthode par séparation et évaluation (branch and bound) [13, 14]. D'autres méthodes telles que la programmation linéaire ou la programmation dynamique sont aussi utilisées couramment [15]. Ce type de méthodes est particulièrement utilisé pour résoudre les problèmes de petites tailles.

II.2.1.1 La méthode de séparation et évaluation

L'algorithme Branch and Bound consiste à placer progressivement les tâches sur les ressources en explorant un arbre de recherche décrivant toutes les combinaisons possibles. Il s'agit de trouver la meilleure configuration donnée de manière à élaguer les branches de l'arbre qui conduisent à de mauvaises solutions. L'algorithme branch and bound effectue une recherche complète de l'espace des solutions d'un problème donné, pour trouver la meilleure solution. [12]
La démarche de l'algorithme Branch and Bound consiste à [16] :

- diviser l'espace de recherche en sous espaces,
- chercher une borne minimale en terme de fonction objectif associée à chaque sous espace de recherche,
- éliminer les mauvais sous-espaces,
- reproduire les étapes précédentes jusqu'à l'obtention de l'optimum global.

II.2.1.2 La programmation dynamique

La programmation dynamique se base sur le principe de Bellman [17] : « Si 'C'est un point qui appartient au chemin optimal entre A et B, alors la portion de ce même chemin allant de A à C est le chemin optimal entre A et C ». C'est une méthode qui consiste donc à construire d'abord les sous chemins optimaux et ensuite par récurrence le chemin optimal pour le problème entier.

Le principe de la méthode dynamique c'est composer une solution optimale du problème en combinant les solutions (optimales) de ses sous problèmes. En pratique : décomposer le problème en des sous-problèmes plus petits, calculer les solutions optimales de tous ces sous problèmes et les garder en mémoire et enfin, calculer la solution optimale à partir des solutions optimales des sous-problèmes [18].

II.2.1.3 La programmation linéaire

De nombreux problèmes pratiques de la recherche opérationnelle peuvent être exprimés sous forme de problèmes de programmation linéaire.

La programmation linéaire est une approche générique basée sur la modélisation mathématique de problèmes d'optimisation combinatoire, où les contraintes, exprimées en inégalités et la fonction objective comme une équation, sont linéaires en ce qui concerne les variables de décision. Donc la programmation linéaire des problèmes détermine la manière d'obtenir le meilleur résultat (comme un profit maximum ou un coût) étant donné une liste d'exigences représentées sous forme d'équation linéaire.

II.2.2 Les méthodes approchées

II.2.2.1 Les métaheuristiques

Les métaheuristiques sont souvent inspirées à partir des systèmes naturels, qu'ils soient pris en physique (cas du recuit simulé), en biologie de l'évolution (cas des algorithmes génétique), ou encore en éthologie (cas des algorithmes de colonie de fourmis ou de l'optimisation par essaim de particules) [19].

Les métaheuristiques constituent une classe de méthodes qui fournissent des solutions de bonne qualité en un temps raisonnable à des problèmes combinatoires réputés difficiles pour lesquels on ne connait pas de méthode classique plus efficace [20]. Elles sont généralement des algorithmes stochastiques itératifs, qui progressent vers un optimum global, c'est à dire l'extremum global d'une fonction en évaluant une certaine fonction objectif. Elles se comportent comme des algorithmes de recherche, tentant d'apprendre les caractéristiques d'un problème à fin d'en trouver une approximation de la meilleure solution d'une manière proche des algorithmes d'approximations.

L'intérêt croissant apporté aux métaheuristiques est tout à fait justifié par le développement des machines avec des capacités calculatoires énormes, ce qui a permis de concevoir des métaheuristiques de plus en plus qui ont fait preuve d'une certaine efficacité lors de la résolution de plusieurs problèmes à caractère NP-difficile [20].

II.2.2.2 Classification des métaheuristiques

Les métaheuristiques sont divisées sur plusieurs classes parmi les on a :

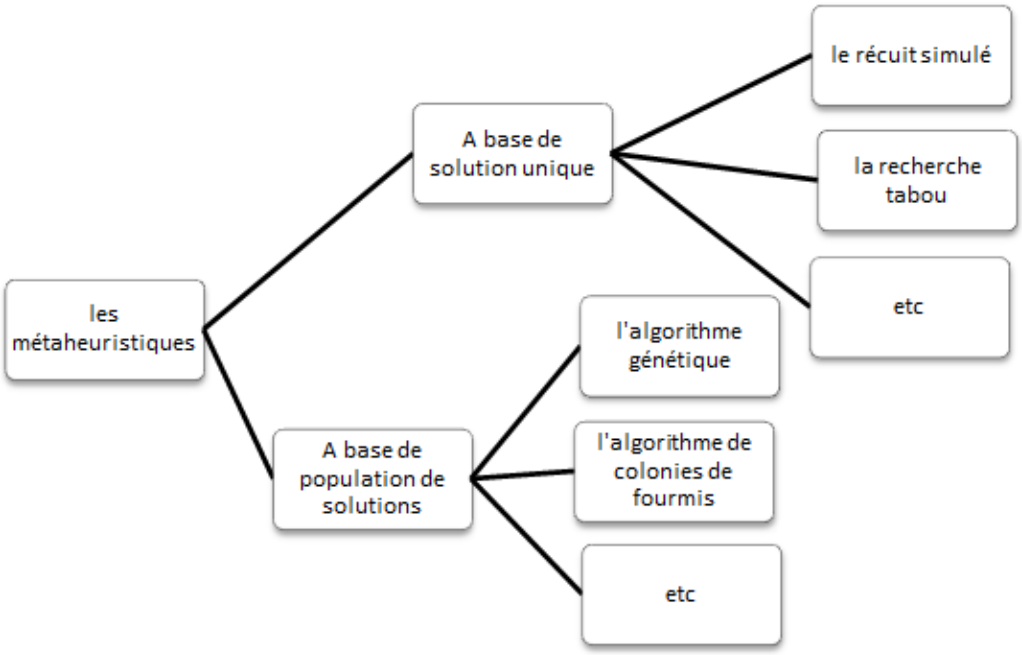


Figure 10: Classification des métaheuristiques.

II.2.2.2.1 Les métaheuristiques basées sur une solution unique

Les métaheuristiques à base de solution unique reposent sur la notion de recherche de voisinage. Elles démarrent d'une solution initiale (générée aléatoirement ou par une précédente méthode d'optimisation) puis, de manière itérative, substituent la solution courante par une autre solution, généralement meilleure, de son voisinage candidat. Le processus de recherche cesse lorsqu'un certain critère dit de continuation n'est plus avéré [21].

Les métaheuristiques à base de solution unique sont plus adaptées à intensifier la recherche dans une région prometteuse

II.2.2.2.2 Les métaheuristiques basées à une population des solutions

Travaillant sur un ensemble de points de l'espace de recherche en commençant avec une population de solution initiale puis de l'améliorer au fur et à mesure des itérations, caractérisées par une tendance à l'exploration et à la diversification.

Ces méthodes sont aptes à générer de « bonnes » solutions, uniformément réparties dans l'espace de recherche. Nous citerons, entre autres, les algorithmes évolutionnaires, les colonies de fourmis et la recherche par dispersion (scatter search) [22].

II.2.2.3 Les heuristiques

Les méthodes dites heuristiques sont des méthodes spécifiques à un problème particulier. Elles nécessitent des connaissances du domaine du problème traité. En fait, ce sont des règles empiriques qui se basent sur l'expérience et les résultats acquis afin d'améliorer les recherches futures [23].

Les Heuristiques permettent de trouver de manière rapide une solution réalisable à un problème donné. Cependant cette solution n'est pas forcément la solution optimale.

L'objectif d'une heuristique est donc de trouver une solution la plus proche possible de celle d'une méthode exacte tout en étant plus rapide [24]. Ces heuristiques ont pour objectif de construire itérativement une solution à un problème d'ordonnancement. Elles consistent à définir des priorités entre les tâches qui sont en attente de traitement sur une machine. Les méthodes les plus couramment utilisées sont des algorithmes dont le principe est de trier les opérations selon une stratégie de décision telle que LPT (Longest Processing Time first), SPT (Shortest Processing Time first) [25], et on distingue :

-L'heuristique SPT (**Shortest Processing Time**): La priorité est donnée aux opérations/commandes qui ont un délai de fabrication le plus court. On parle aussi de (Traitement par ordre croissant du temps opératoire).

-L'heuristique LPT (**Longest Processing Time**): Tout d'abord, Graham (1969) propose l'heuristique LPT .Cette méthode commence par trier les tâches à ordonnancer en ordre décroissant du temps de traitement. Par la suite, elle assigne chaque tâche, dans l'ordre, à la première machine disponible.

-L'heuristique EDD (Earliest Due Date):

La priorité est donnée aux opérations/commandes dont la date de fin promise est la plus proche (Traitement par ordre croissant du délai de livraison).

-L'heuristique FCFS (First come, first served):

La priorité est donnée aux opérations/commandes selon leur ordre d'arrivée.

-L'heuristique SRPT (Shortest Remaining Processing Time):

Cette règle, servant à lancer la tâche ayant la plus courte durée de travail restant à exécuter, est très utilisée pour minimiser les encours et dans le cas des problèmes d'ordonnancement préemptifs [26].

II.3 L'atelier job shop

Le problème du job shop, ou problème d'atelier à cheminements multiples, consiste généralement à organiser la réalisation d'un ensemble de travaux ou jobs sur un ensemble de machines, où chaque job est composé d'une séquence d'opérations non préemptives, devant être exécutées sur ces machines selon un ordre prédéfini propre à lui qui n'est pas nécessairement le même pour tous les jobs [27].

Les premiers résultats théoriques du job shop, concernant la résolution d'un problème a deux machines et plus tard à trois machine, revient aux années cinquante. La première définition formelle de cet atelier a paru dans l'ouvrage de Muth et Thompson 1963. Certains problèmes job shop, proposes à cette époque, ont été résolus dans les années soixante-dix. En revanche, il a fallu attendre les années quatre-vingt pour d'autres. Jusqu'à ce jour cet atelier reste le contre d'intérêt d'un nombre important de chercheurs [28].

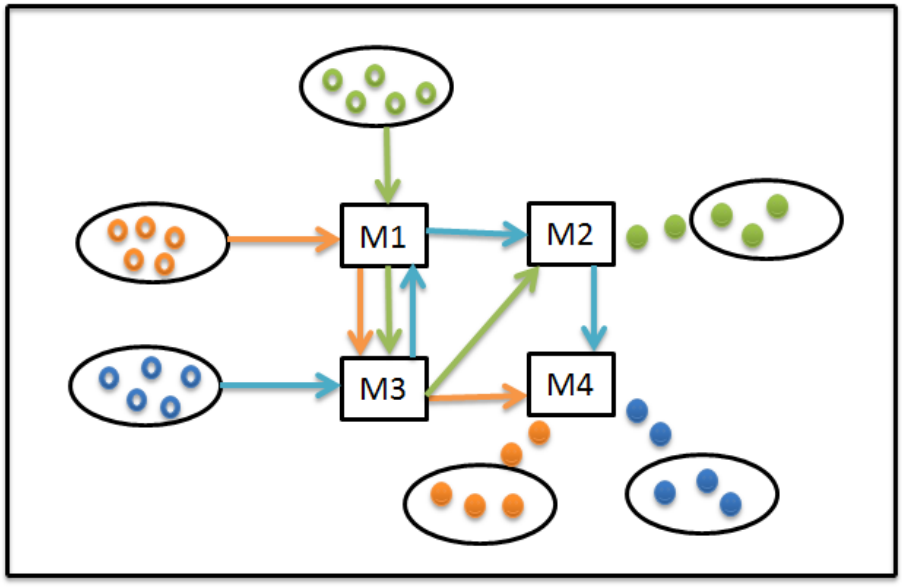


Figure 11: ordonnancement d'une séquence de jobs en Job Shop.

Table 1: Gamme opératoire des jobs.

	1	2	3	4	1	2	3	4	1	2	3	4
J_1	M_1	M_2	M_3		10	8	4		$O_{1,1}$	$O_{2,1}$	$O_{3,1}$	
J_2	M_2	M_1	M_4	M_3	8	3	5	6	$O_{2,2}$	$O_{1,2}$	$O_{4,2}$	$O_{3,2}$
J_3	M_1	M_2	M_4		4	7	3		$O_{1,3}$	$O_{2,3}$	$O_{4,3}$	
	Machines				Durée opératoire (Pij)				Operations			

Le job shop flexible représente une généralisation du problème du job shop, du fait qu'une opération donnée peut être réalisée par une ou plusieurs ressources et possède une durée de traitement dépendant de la ressource utilisée. Une solution au problème job shop flexible ne consiste plus uniquement à trouver une séquence des opérations sur chacune des ressources et à leur fixer une date de début, mais également à déterminer une affectation de chaque opération à l'une de ses ressources potentielles, de telle sorte que la durée totale de l'ordonnancement est minimisée. Ce problème est NP-difficile puisque le problème classique l'est aussi [29].

II.3.1 Problématique de l'ordonnancement en Job Shop

Nous nous intéressons plus spécifiquement aux méthodes permettant de résoudre des problèmes d'ordonnancement de n ordres de fabrication (ou jobs) sur m (m>2) machines.

Les problèmes d'ordonnancement en job shop sont des problèmes classiques en littérature d'ordonnancement se compose d'un ensemble fini de travaux n devant traité sur un ensemble fini de machines m et est dénoté comme problème n × m, sachant que chaque machine ne peut travailler que sur un seul job à la fois. Les notations couramment employées dans les problèmes d'ordonnancement sont les suivantes [30] :

• La date de début au plus tôt du job Ji est notée ri (release date). Cette date correspond à un minorant de la date de disponibilité du job Ji.

• di et Ci sont respectivement la date de fin au plus tard souhaitée (due date) et la date effective de fin de fabrication (completion date) du job Ji.

On peut décomposer les problèmes d'ordonnancement en deux phases, une première phase qui tente d'attribuer des caractéristiques temporelles aux opérations à effectuer et une deuxième phase d'attribution des ressources aux différentes opérations [31] [30].

• La première phase est décrite par des contraintes potentielles qui regroupent les contraintes de précédence entre tâches (aucune panne de machine dans tout le processus d'ordonnancement), prenant en compte la succession des opérations de la gamme opératoire (la tâche i doit démarrer après la tâche j) et les contraintes d'une tâche dans le temps(le temps transport entre les machines est zéro).

• La deuxième phase est décrite par les contraintes disjonctives qui expriment le fait que deux opérations utilisant une même ressource, ne peuvent avoir d'intervalle de temps commun [32] [30].

II.3.2 Représentation graphique de job shop

Pour traiter n'importe problème de job shop de telle complexité nécessite une représentation graphique claire et précise, plusieurs manières de modélisation conduisent à choisir le mode de représentation convenable, de telle sort la modélisation par un graphe disjonctif qui est la plus classique et la plus répandue, la modélisation par diagramme de Gantt, la modélisation par méthode de Pert …etc.

II.3.2.1 Le graphe disjonctif

Le graphe disjonctif est une modélisation parmi les modélisations graphiques utilisée pour représenter les problèmes d'ordonnancement de type job shop. Elle est proposée par Roy et Susmann en 1964, pour faciliter la résolution des problèmes d'ordonnancement de type job shop, cette représentation constitue d'un ensemble de sommets relié entre eux par des arcs.

Soit G = (N;A1∪A2) ce graphe :

N : est l'ensemble des sommets, chaque opération correspond à un sommet, ainsi qu'une source (S) représentant le début de tous les jobs et un puits(P) représentant leur fin, deux opérations fictifs [33].

A1 : est l'ensemble des arcs conjonctifs représentant les contraintes d'enchaînement des opérations d'une même tâche (gammes opératoires). Pour un arc (ij, ik) de la partie conjonctive, on a [33] :

$$tij - tik \geq pi,k \quad \forall \ (ij,ik) \in C$$

On aura donc un arc entre tous les sommets (i, j), (i, j+1) pour i = 1…n et j = 1…ni-1 ; (n : nombre de tâches, ni : nombre d'opérations de la tâche i). De plus, il existe des arcs entre S et tous les sommets (i, 1), et d'autres arcs entre les sommets (i, ni) et le puits P [33].

A2 : l'ensemble des arcs disjonctifs associés aux conflits d'utilisation d'une machine. Pour un arc (ij, ik) de la partie disjonctive, on a [34] :

tjk - tik ≥ pik ou tik - tjk ≥ pjk ∀ (ik,jk) ∈ D

Que l'on modélise par un arc et un arc retour entre les sommets (tjk, tik) représentant deux opérations utilisant la même machine.

Une orientation de chaque arc disjonctif (de sorte à avoir un graphe acyclique, ie. Sans circuit) permet d'obtenir une solution réalisable du problème d'ordonnancement. Le graphe disjonctif est dit alors arbitré (arbitrated) [35]. La longueur du plus long chemin allant de la source au puits est désignée la valeur de makespan.

La figure correspond au graphe disjonctif d'un atelier composé de trois jobs et trois machines, tel que [36] :

-N={$O_{1,1}$,$O_{1,2}$,$O_{1,3}$, $O_{2,1}$,$O_{2,2}$,$O_{2,3}$,$O_{3,1}$,$O_{3,2}$,$O_{3,3}$ }.

-Le sommet S (début) précède tous les sommets du graphe G et P (fin) succède à tous, et tous les sommets correspondant à des opérations sont pondérés par leur temps opératoire.

Rappelons que les arcs conjonctifs représentent les contraintes de précédence entre les opérations d'un même job (par exemple : $O_{1,1}$, $O_{1,2}$ et $O_{1,3}$).

De plus, la clique cyclique entre les sommets $O_{1,1}$, $O_{1,2}$ et $O_{1,3}$ signifie un conflit d'utilisation d'une même ressource.

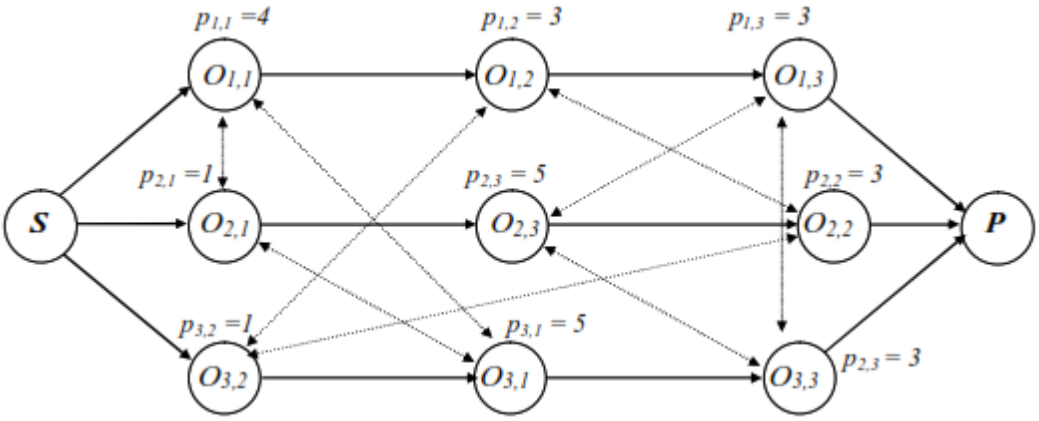


Figure 12: Le graphe disjonctif du problème de Job Shop 3x3 [36].

II.3.2.2 Le diagramme de Gant

Le diagramme de Gantt tient son nom de son créateur Henry Laurence Gantt (1861-1919), ingénieur en mécanique et consultant en management. Il est surtout connu pour avoir mis au point en 1910 son célèbre diagramme très utilisé en gestion de projets. Il fut créé en 1917. Le but de ce diagramme est de représenter de manière claire et rapide la solution à un problème d'ordonnancement. Plusieurs variantes ont vu le jour depuis mais, au sens large, il se représente par un ensemble de lignes horizontales tracées dans un graphe avec le temps en abscisses. Chaque ligne est composée de différents rectangles qui sont pour chacun la représentation du temps nécessaire à la réalisation d'un travail : le temps de réalisation est proportionnel à la longueur du rectangle. Dans le cadre d'un problème d'ordonnancement, le diagramme de Gantt prend en ordonnée les différentes ressources utilisées et donne une représentation sous forme de diagramme de Gantt d'un problème du job-shop [37].

Deux types de diagramme de Gantt sont utilisés : Gantt ressources et Gantt tâches (voir la figure 13) [38] :

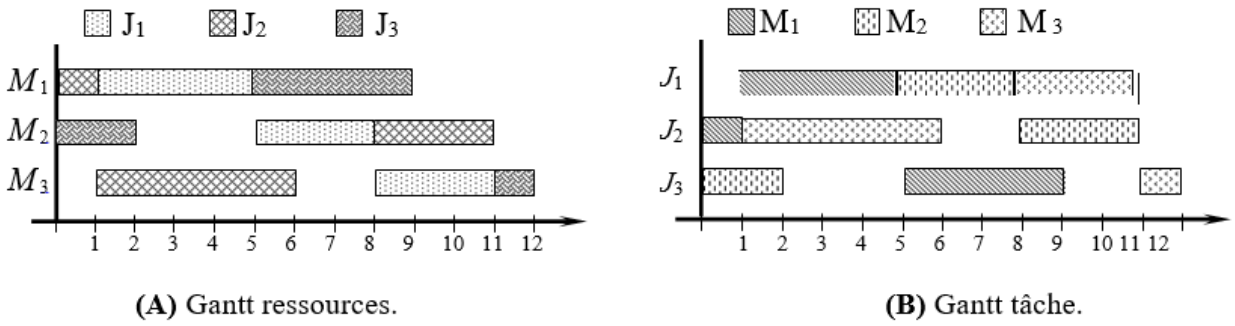


Figure 13: Diagrammes de Gantt pour un problème de Job Shop du 3x3 [38].

 Le diagramme de Gantt ressources, 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 tâche, 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 où la tâche est en attente des ressources.

II.4 Conclusion

Le problème du job shop, consiste à organiser la réalisation d'un ensemble de travaux ou jobs sur un ensemble de machines, chaque job a une séquence d'opérations non préemptives, les ordres d'exécutions sur les machines ne sont pas les mêmes pour tous les jobs. Pour résoudre un problème de job shop nécessite de connaitre sa classe de complexité au premier pour choisir la méthode adéquate ainsi que la modélisation graphique qui nous aide dans le traitement de ce problème.

L'objectif le plus considéré dans le cas d'un atelier à cheminements multiples à savoir trouver une séquence de tâches sur les machines qui minimise le temps total de production.

III Chapitre 3 : application des heuristiques sur problème job shop

III.1 Introduction

Dans l'industrie se trouve plusieurs système de production, nous avons intéressé à étudier un problème d'ordonnancement d'un système de type job shop sur lequel nous essayons d'appliquer certains heuristiques et voir leurs performances dans la résolution.

Dans ce chapitre nous allons présenter notre système à étudier, et définir notre problème de job shop ainsi que les heuristiques utilisées pour la résolution de notre problème ainsi que nos logiciels de simulation et voir les résultats obtenus.

III.2 Description de système

Nous allons étudier dans ce mémoire un système de production de type job shop inspiré d'un system réel qui se trouve au niveau de laboratoire de productique MELT (université de Tlemcen), qu'ils sont pareils de principe de fonctionnement.

Ce système est composé de quatre machines M1, M2, M3 et M4 installés sous forme d'un rectangle et les produits à traiter se trouvent dans l'espace de stock d'entrées, les produits sont transportés dans un sens unidirectionnel par un tapie roulant automatique devant les différents machines. Un bras manipulateur est installé devant chaque machine qui sert à déplacer les pièces du tapie vers les machines et l'inverse [39].

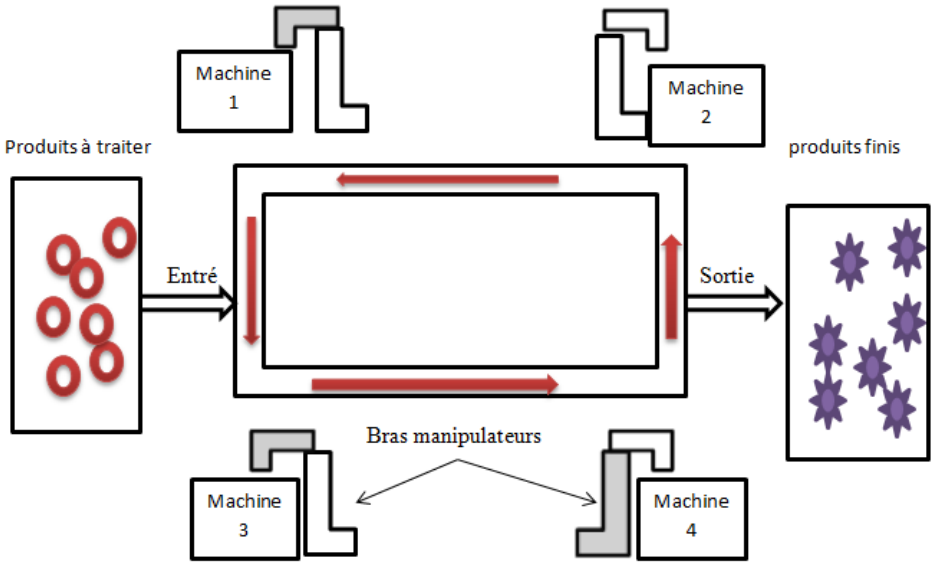


Figure 14: le système à étudier.

III.3 Formalisation de problème job shop

Soit un problème de job shop. Les notations suivantes sont introduites pour décrire le problème. Soit M l'ensemble des m machines M_1, M_2,…, M_n et J l'ensemble des n jobs J_1, J_2,…, J_n . On note O l'ensemble des o opérations O= {O_1, O_2,…, O_n} décrivant la réalisation des jobs sur les machines.

Les jobs sont décrits par des gammes. Une gamme définit l'ordre, la durée et la séquence des machines où sont traitées les opérations d'un job. La gamme définit pour chaque job J_i une suite K_i de k_i opérations dont la première est noté d_i : $K_i = (O_{di}, O_{di+1},\ldots, O_{di+ki-1})$. Les suites K_i sont disjointes, c'est-à-dire qu'une opération n'apparait que dans une seul suite K_i. la suite des opérations de K_i appelée ordre d'opération de la gamme. Un job ne peut commencer le traitement d'une opération que quand l'opération précédente dans la gamme est terminée .On appelle contrainte de précédence cette contrainte et on note $A = (O_i, O_{i+1})$ l'ensemble des opérations successives soumise à une contrainte de précédence direct [40].

Une opération O_i doit être réalisée pendant un temps P_i sur une machine m_i. Cette opération doit être réalisée sans préemption.

On note E_k ou K € M l'ensemble des opérations devant être traitées sur la machine M_k. Le nombre d'opération à traiter sur la machine M_k et donc $e_k = |E_k|$. Une machine ne peut traiter qu'une seul opération à la fois .les opérations de E_k doivent donc être ordonnées, on dit qu'il y a disjonction entre elles car les intervalles de temps pendant lesquels la machine est occupée sont disjoints [41].

On notera les données utilisées dans un problème job shop come suite :

N : Nombre totale de jobs.

m : Nombre totale de machines.

I : indice du ième job.

j : Indice de la jème opération du job j_i.

J_{io}: Nombre totale des opérations dans le job j_i.

Oij : ***jéme*** Opération du job j_i.

$\boldsymbol{\Omega}_{ij}$: Ensemble des machines disponibles pour l'opération O_{ij}.

P_{ijk}: temps d'exécution de l'opération O_{ij} sur la machine k.

S_{ijk}: Date de début de l'opération O_{ij} sur la machine k.

E_{ijk}: Date de fin de l'opération O_{ij} sur la machine k.

L=Σ$\boldsymbol{Jion1}$: somme de toute les opérations dans tous les jobs.

La modélisation mathématique d'un job shop classique [42].

La fonction Objectif :

Minimise [Max (C_{1J1}, C_{2J2},…, C_{nJn})] (1)

Sous contraintes **:**

$C_{ij} - S_{ij} - P_{ij} = 0 \quad \forall i,$ (2)

$C_{i'j'} - C_{ij} + H\,(1 - Y_{iji'j'}) \geq P_{i'j'} \quad \forall (i,j), \forall (i',j'): O_{ij} \in N_k , O_{i'j'} \in N_k$ (3)

$C_{ij} - C_{i'j'} + (Y_{iji'j'}) \geq P_{i'j'} \quad \forall (i,j), \forall (i',j'): O_{ij} \in N_k , O_{i'j'} \in N_k$ (4)

$S_{ij} \geq 0 \quad \forall i,j$ (5)

$S_{ij+1} \geq C_{ij} \quad \forall i,j=1,\ldots,i-1$ (6)

Signification des équations:

-Contrainte 2 : contrainte de précédence entre les opérations d'un même job : Une opération ne peut commencer à être exécutée avant la fin du traitement de l'opération précédente du même job.

-Contraintes 3 ,4 : contrainte de partage de ressource. Une machine ne peut traiter qu'une seule opération à la fois,

-Contrainte 5 : cette contrainte signifie que la date de début de chaque opération doit être positive ou nulle.

-Contrainte 6 : cette contrainte impose que la date de début d'une opération suivante doit être supérieure ou égale au date de fin de l'opération précédente de même job.

III.4 Contraintes et objectifs

Le problème de job shop présente des hypothèses à respecter et des conditions particulières sur l'environnement d'exécution concernant à la fois les ressources et les contraintes de temps. Nous allons par la suite citer les hypothèses liées au job shop [43].

-les machines ne sont pas identiques : chaque machine est unique. Lors de l'affectation, aucune question de type "sur quelle machine une telle opération doit s'effectuer "ne peut se poser.

-les opérations d'un même job s'exécutent séparément sur les machines en respectant l'ordre des séquences O_i.

-la préemption des opérations n'est pas permise : une fois commencée l'exécution d'une opération ne peut pas être interrompue.

-toutes les données du job shop sont connues d'avance : il s'agit alors du job shop statique (en particulier, il ne doit pas y avoir des nouveaux jobs lors de l'exécution des anciens.

-Les machines ont une disponibilité infinie. Elles sont supposées fiables et ne subissent aucune panne pouvant perturber ou arrêter l'exécution des opérations.

-Les temps de montage et de démontage sont inclus dans les temps de traitement.

-Les temps de transport sont négligeables.

III.5 Les heuristiques utilisées

Les heuristiques sont parmi les méthodes approchées utiliser pour résoudre un problème d'ordonnancement, elles ont pour objectif de construire itérativement une solution à un problème d'ordonnancement. Elles consistent à définir des priorités entre les tâches qui sont en attente de traitement sur une machine. Les méthodes les plus couramment utilisées sont des algorithmes dont le principe est de trier les opérations selon une stratégie de décision telle que LPT (Longest Processing Time first), SPT (Shortest Processing Time first), etc.

III.5.1 L'heuristique SPT (shortest Processing Time)

Est une règle classique de priorité, qui consiste à ordonner les taches dans un ordre croissant de passage sur les machines par rapport à ses temps opératoires d'exécution sur les machines [44].

Table 2: exemple des taches et leurs temps opératoires pour SPT.

	M1	M2	M3	
J1	15 (1)	4 (3)	7 (2)	26
J2	10 (2)	11 (1)	6 (3)	27
J3	12 (1)	8 (2)	2 (3)	22

D'après l'heuristique SPT la séquence est :

J3 ⟶ J1 ⟶ J2

Cmax=65j

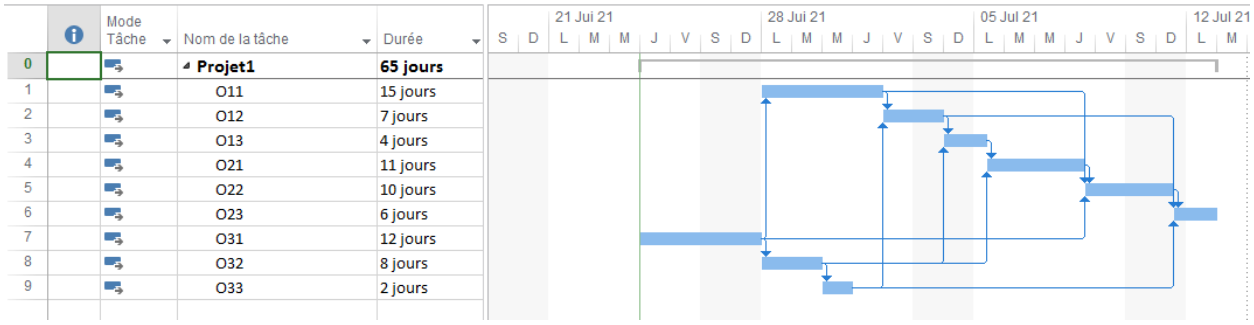


Figure 15: diagramme de Gantt pour l'exemple de SPT.

III.5.2 L'heuristique LPT (Longest Processing Time)

est une règle classique de priorité, qui consiste à ordonner les taches dans un ordre décroissant de passage sur les machines par rapport à ses temps opératoires d'exécution sur ces machines [45].

Table 3: exemple des taches et leurs temps opératoires pour LPT.

	M1	M2	M3	
J1	15 (1)	4 (3)	7 (2)	26
J2	10 (2)	11 (1)	6 (3)	27
J3	12 (1)	8 (2)	2 (3)	22

D'après l'heuristique LPT la séquence est :

J2 ⟶ J1 ⟶ J3

Cmax=58j

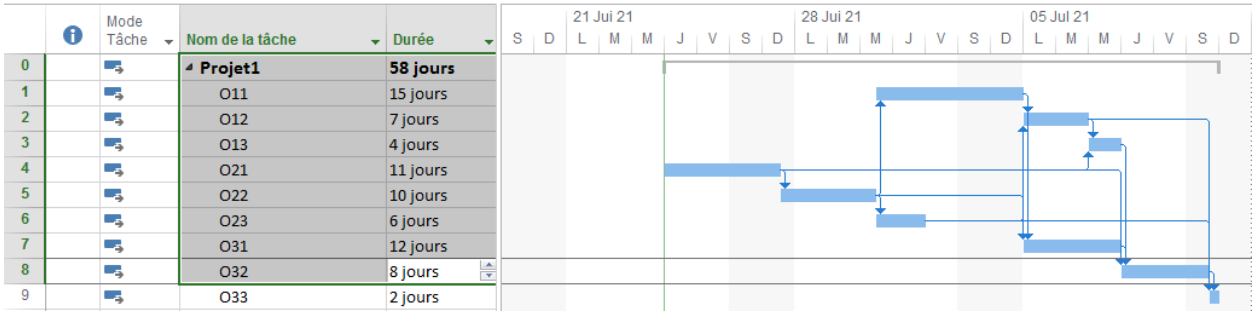


Figure 16: diagramme de Gantt pour l'exemple de LPT.

III.5.3 L'heuristique shifting Bottleneck

L'heuristique Shifting Bottleneck est une procédure souvent utilisée pour minimiser le makespan dans un atelier de type job shop pour lequel les ressources de traitement sont critiques. Cette heuristique est destinée à minimiser l'effet de goulet d'étranglement (bottleneck) en ordonnançant d'abord les ressources qui causent potentiellement le plus de retard [46]. (Cette heuristique est bien détaillée dans le mémoire de cette référence [46]).

Sa principe de fonctionnement est défini par l'organigramme suivant :

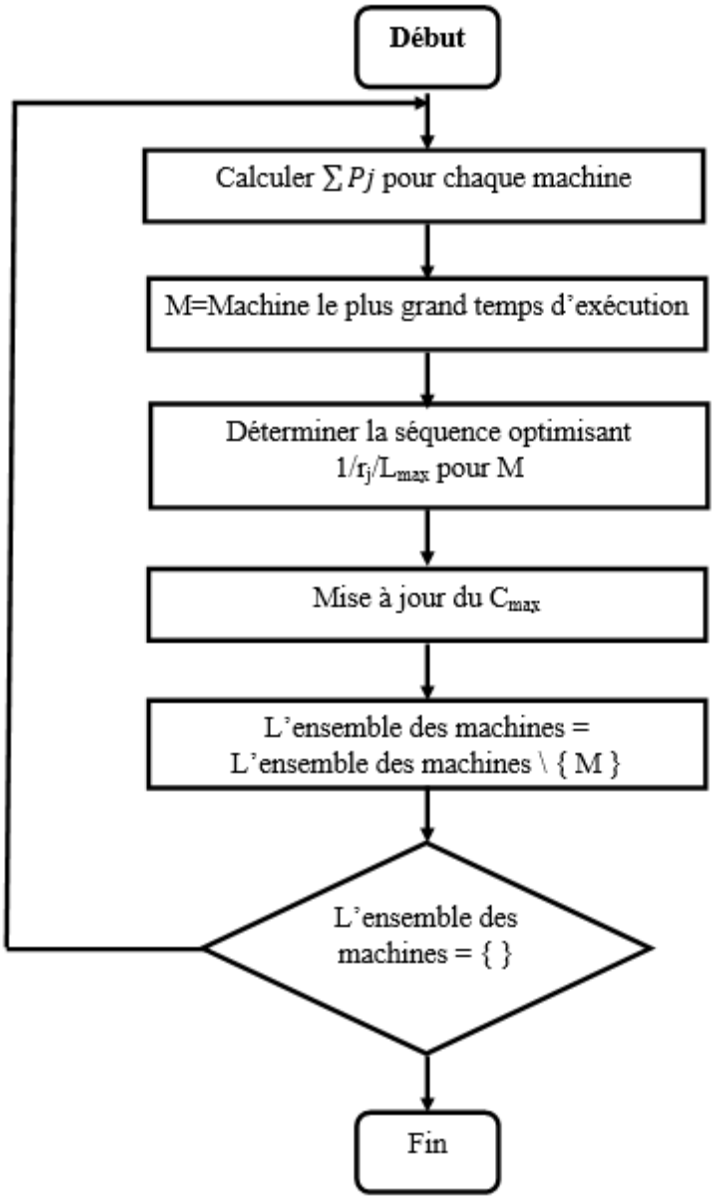


Figure 16 : organigramme de principe de fonctionnement shifting Bottleneck [46].

Table 4: exemple des taches et leurs temps opératoires Shifting Bottleneck.

	M1	M2	M3
J1	15 (1)	4 (3)	7 (2)
J2	10 (2)	11 (1)	6 (3)
J3	12 (1)	8 (2)	2 (3)

D'après l'heuristique shifting bottleneck le séquencement est :

M1 : J3 ⟶ J2 ⟶ J1

M2 : J2 ⟶ J3 ⟶ J1

M3 : J3 ⟶ J2 ⟶ J1

Cmax=48j

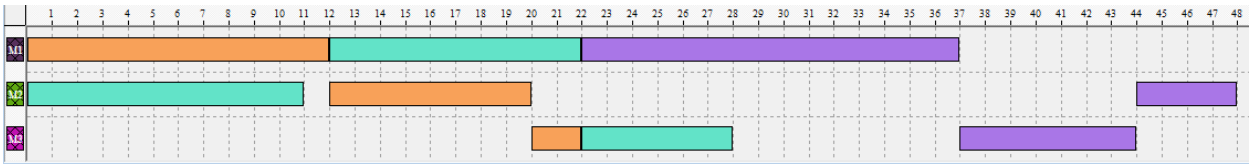


Figure 17: diagramme de Gantt pour l'exemple de shifting bottleneck.

III.5.4 SPT (machine disponibilité)

Est une règle de priorité qui consiste à ordonner les jobs en un ordre croissant par rapport au temps opératoires d'exécution sur les machines. Si l'une des machines de job 1 est disponible pour traiter le job 2 cette job n'attend pas la précédente job jusqu'à sortir de système et termine l'exécution, elle commence le traitement sur cette machine.

Table 5: exemple des taches et leurs temps opératoires SPT (disponibilité machine).

	M1	M2	M3	
J1	15 (1)	4 (3)	7 (2)	26
J2	10 (2)	11 (1)	6 (3)	27
J3	12 (1)	8 (2)	2 (3)	22

M1 : J3 ⟶ J2 ⟶ J1

M2 : J2 ⟶ J3 ⟶ J1

M3 : J3 ⟶ J2 ⟶ J1

Cmax=48j.

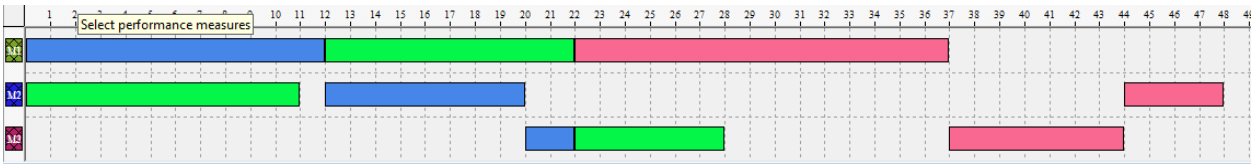


Figure 18: diagramme de Gantt pour l'exemple de SPT (machine disponibilité).

III.5.5 LPT (machine disponibilité)

Est une règle de priorité qui consiste à ordonner les jobs en un ordre décroissant par rapport au temps opératoires d'exécution sur les machines. Si l'une des machines de job 1 est disponible pour traiter le job 2 cette job n'attend pas la précédente job jusqu'à sortir de système et termine l'exécution, elle commence le traitement sur cette machine.

Table 6: exemple des taches et leurs temps opératoires LPT (disponibilité machine).

	M1	M2	M3	
J1	15 (1)	4 (3)	7 (2)	26
J2	10 (2)	11 (1)	6 (3)	27
J3	12 (1)	8 (2)	2 (3)	22

D'après la règle LPT (avec disponibilité machine) le séquencement est :

M1 : J1 ⟶ J3 ⟶ J2

M2 : J2 ⟶ J1 ⟶ J3

M3 : J1 ⟶ J3 ⟶ J2

Cmax=43j.

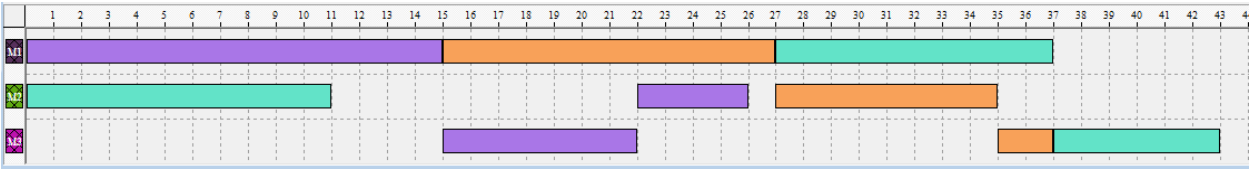


Figure 19: diagramme de Gantt pour l'exemple de LPT (machine disponibilité).

III.6 Logiciels de simulation

III.6.1 Ms Project

MS Project permet la planification d'un projet : il est possible à tout moment créer des tâches et des jalons, définir les liens entre chaque tâche, les hiérarchiser. MS Project a également la capacité d'estimer la durée ainsi que la charge de travail nécessaire pour accomplir une tâche définie. Microsoft Project permet aussi la création de modèles qui permet à l'utilisateur de respecter une méthodologie ou un processus quelconque. Le projet peut être représenté graphiquement de différentes manières : diagramme de Gantt, réseau des tâches… Le pilotage du projet est possible par de multiples façons telles que la définition de la planification initiale, la saisie de l'avancement des tâches ou bien la replanification [47].

Parmi ses fonctionnalités on trouve :

- Planification et pilotage des projets.
- Gestion des ressources.
- Gestion des coûts.
- Analyse et communication des informations des projets.

III.6.2 LEKIN

LEKIN est un système de planification développé à la Stern School of Business, NYU. Les principales parties du système ont été conçues et codées par des étudiants de l'Université de Columbia. LEKIN® a été créé comme un outil pédagogique dans le but principal d'initier les étudiants à la théorie de l'ordonnancement et à ses applications. En plus de cela, l'extensibilité du système permet (et encourage) de l'utiliser dans le développement d'algorithmes. Le projet a été dirigé par le professeur Michael L. Pinedo, le professeur Xiuli Chao et le professeur Joseph Leung. Ce développement a été partiellement soutenu par la National Science Fondation [48].

Parmi ses caractéristiques on trouve [48] :

- 6 environnements d'espace de travail de base : machine unique, machines parallèles, atelier de flux, atelier de flux flexible, atelier de travail et atelier de travail flexible.
- Diverses règles de répartition et heuristiques.
- Diagramme de Gantt avec prise en charge du glisser-déposer.
- Outil graphique pour l'analyse comparative de différents horaires.

III.7 Simulation et résultats expérimentaux

Pour étudier notre système nous avons effectué certains exemples d'application pour des problèmes de différentes tailles on utilisant certaines heuristiques pour voir ses performances par la comparaison entre les résultats obtenus de Cmax. On va appliquer les règles d'ordonnancement SPT et LPT sur notre système et sur les machines ainsi que l'heuristique shifting bottleneck, sur des exemples de petites tailles, moyennes tailles et grandes tailles.

Nous allons utiliser les deux logiciels Ms project et Lekin pour résoudre les problèmes de notre système ensuite on va comparer entre les résultats obtenus.

Nous avons étudié cinq exemples de deux catégories pour chaque taille de problème, ces exemples sont générés de manière aléatoire avec des gammes opératoires aléatoires avec les durées d'exécutions des jobs dans les machines sont pris de l'intervalle [0,30].

III.7.1 Exemples de petites tailles

Nous allons traiter cinq problèmes de trois jobs avec quatre machines. Le tableau suivant montre les résultats obtenus par l'utilisation des trois heuristiques.

Table 7: exemples de 3 jobs et 4 machines.

les exemples	Spt système	Lpt systeme	Spt machine	Lpt machine	Shiffting bottleneck
1	158	210	116	109	87
2	159	159	74	95	75
3	149	144	70	71	70
4	107	87	87	87	74
5	148	130	118	74	81
Moyen Cmax	144	146	93	88	78

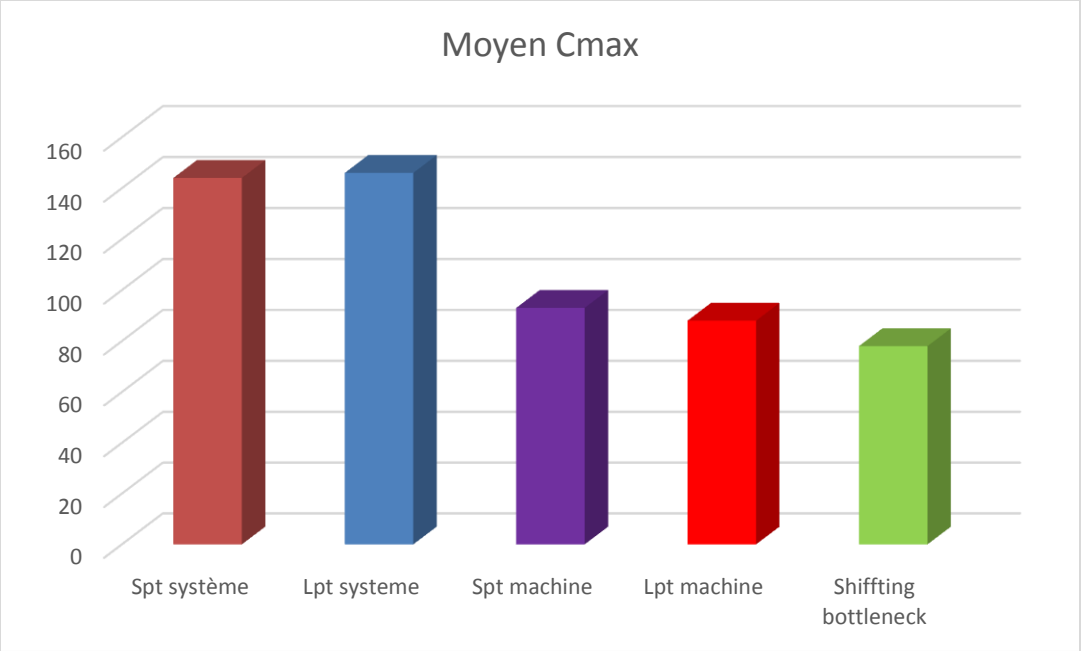


Figure 20: Résultats obtenus pour le problème 3x4.

Nous allons traiter cinq problèmes de quatre jobs avec quatre machines. Le tableau suivant montre les résultats obtenus par l'utilisation des trois heuristiques.

Table 8: exemples de 4 jobs et 4 machines.

les exemples	Spt système	Lpt systeme	Spt machine	Lpt machine	Shiffting bottleneck
1	243	211	127	137	119
2	180	178	110	94	116
3	202	176	128	107	105
4	159	175	122	103	83
5	157	115	101	90	85
Moyen Cmax	188	171	118	107	102

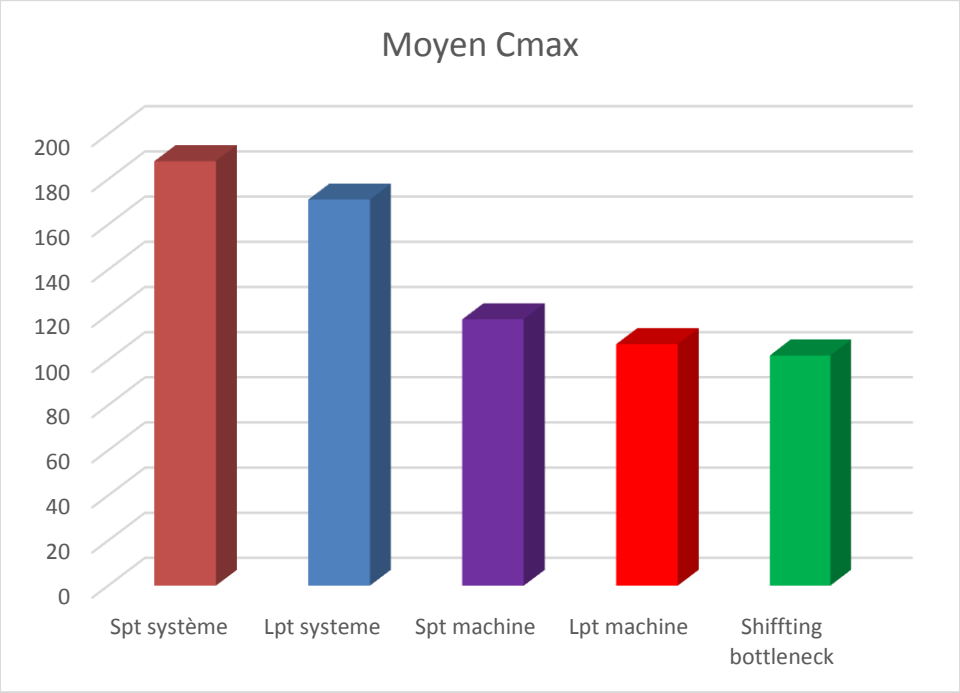


Figure 21: Résultats obtenus pour le problème 4x4.

D'après les résultats obtenus par l'application des heuristiques sur les problèmes de petites tailles nous avons remarqué que l'heuristique shifting bottleneck donne le meilleur résultat de Cmax. Nous pouvons remarquer aussi que la règle LPT machine est meilleur que SPT machine, et les résultats de ces deux méthodes sont beaucoup plus satisfaisants que les résultats de SPT et LPT système.

III.7.2 Exemples de moyennes tailles

Nous allons traiter cinq problèmes de huit jobs avec quatre machines. Le tableau suivant montre les résultats obtenus par l'utilisation des trois heuristiques.

Table 9: exemples de 8 jobs et 4 machines.

les exemples	Spt système	Lpt systeme	Spt machine	Lpt machine	Shiffting bottleneck
1	324	350	205	133	146
2	376	362	228	156	198
3	338	336	249	163	171
4	334	379	165	155	152
5	388	352	218	159	149
Moyen Cmax	352	356	213	154	164

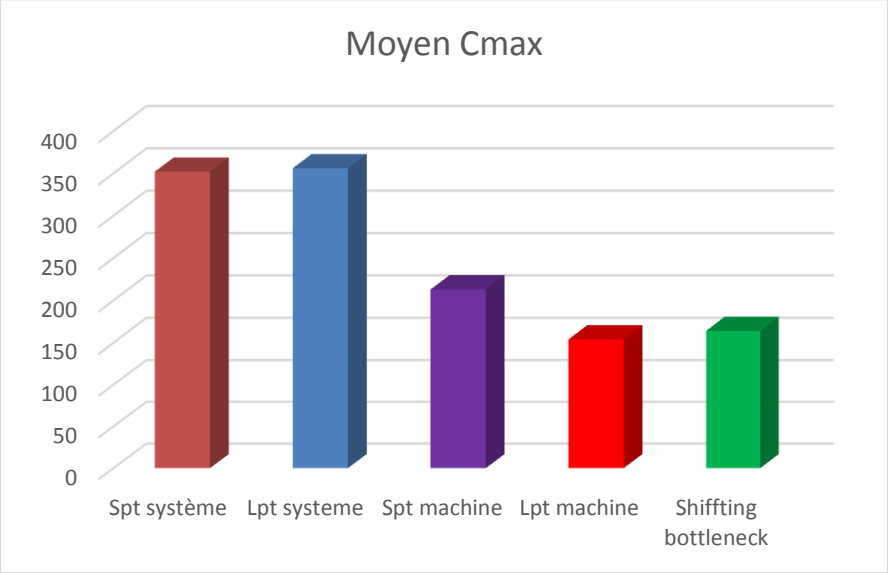


Figure 22: Résultats obtenus pour le problème 8x4.

Nous allons traiter cinq problèmes de dix jobs avec quatre machines. Le tableau suivant montre les résultats obtenus par l'utilisation des trois heuristiques.

Table 10: exemples de 10 jobs et 4 machines.

les exemples	Spt système	Lpt systeme	Spt machine	Lpt machine	Shiffting bottleneck
1	362	404	235	155	218
2	399	376	249	195	227
3	455	470	197	179	190
4	411	349	230	159	175
5	398	505	293	228	233
Moyen Cmax	405	421	241	184	209

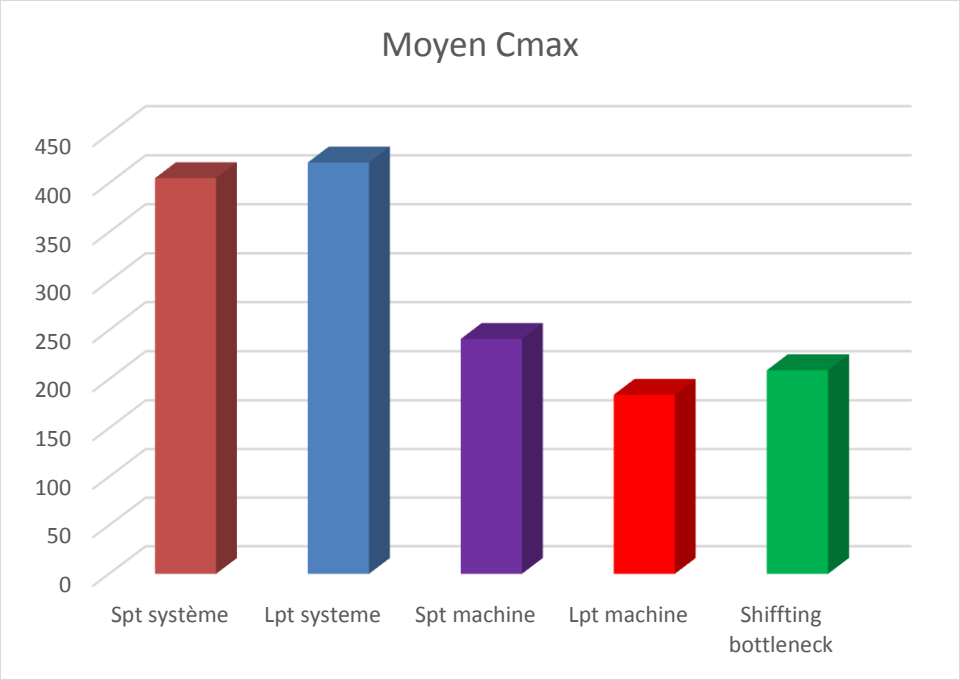


Figure 23: Résultats obtenus pour le problème 10x4.

D'après les résultats obtenus par l'application des cinq heuristiques sur les problèmes de moyennes tailles nous avons remarqué que la règle LPT (avec disponibilité de machine) donne le meilleur résultat de Cmax, ainsi que les résultats fournis par LPT et SPT de système sont beaucoup loin de la solution optimal.au tant que la SPT système est meilleur que LPT système.

III.7.3 Exemples de grandes tailles

Nous allons traiter cinq problèmes de 20 jobs avec quatre machines. Le tableau suivant montre les résultats obtenus par l'utilisation des trois heuristiques.

Table 11: exemples de 20 jobs et 4 machines.

les exemples	Spt système	Lpt systeme	Spt machine	Lpt machine	Shiffting bottleneck
1	//	//	436	336	//
2	//	//	436	367	//
3	//	//	559	407	//
4	//	//	467	350	//
5	//	//	407	338	//
Moyen Cmax	//	//	461	360	//

III.7.4 Interprétation

D'après les résultats montrés dans le tableau 5 nous avons remarqué que la règle LPT (machine avec disponibilité) donne le meilleur résultat de Cmax. Les exemples de 20 jobs ne sont pas réalisables par l'heuristique shifting bottleneck, et elles prennent un très grand temps dans le système.

Selon les résultats précédentes on peut dire que l'heuristique shifting bottleneck donne des résultats satisfaisants de Cmax par rapport les règles SPT et LPT de système et de machines avec disponibilité pour les problèmes d'ordonnancement de petits tailles dans un système d'atelier de type job shop. Ainsi que la règle LPT (machine avec disponibilité) fournis des meilleurs résultats par rapport l'heuristique shifting bottleneck et les autres règles pour les problèmes d'ordonnancement de moyennes et grandes tailles dans un système d'atelier de type job shop.

III.8 Conclusion

Dans ce chapitre nous avons présenté dans la première partie notre système à étudier, ainsi que notre problème job shop, la formulation de problème et ses contraintes et objectives. Nous avons cité par la suite les heuristiques SPT, LPT et shifting bottleneck utilisées pour la résolution des problèmes ainsi que les logiciels de simulation Ms project et Lekin.

Dans la deuxième partie nous avons présenté la simulation et les résultats obtenus par chaque heuristique appliquée sur les différents exemples et nous avons fait une comparaison entre ces résultats en se basant sur la valeur de Cmax pour bien étudier les performances de ces règles de priorités.

IV Conclusion général

Dans notre travail nous avons étudié un problème d'ordonnancement d'un système de production type job shop par l'application des heuristiques différentes à fin de trouver une solution près de l'optimal de makespan. Les problèmes de grandes tailles d'atelier job shop c'est-à-dire un grand nombre de jobs et de machines sont des problèmes complexes et appartient à la classe NP-difficile, le nombre de solutions possibles est très grand.

La résolution d'un problème NP-difficile de type job shop nous avons poussé pour utiliser des heuristiques et les metaheuristique car le nombre de solutions possibles est très grand et ces méthodes fournir des résultats satisfaisants, où l'application des méthodes exactes sera non réaliste.

Parmi les heuristiques nous avons utilisé les règles SPT, LPT ainsi que l'heuristique shifting bottleneck, nous avons appliqué ces trois heuristique sur des différentes problèmes d'atelier job shop où chaque problème se composer de même nombre de jobs, de machines et même séquence des taches.

Nous avons utilisé logiciel Ms Project pour la simulation des problèmes et voir les résultats obtenus par les trois heuristiques puis les comparer selon la valeur de makespan.

D'après les résultats obtenus nous avons remarqué que l'heuristique shifting bottleneck fourni des résultats satisfaisants pour les problèmes de petites tailles, et pour les problèmes de moyennes et grandes tailles l'heuristique LPT (avec disponibilité de machine) donne les meilleurs résultats.

Perspectives :

Pour enrichir ce travail nous avons des perspectives comme :

- Faire même étude sur ce thème on utilisant d'autres règles de priorités.
- Appliquer les metaheuristiques et voir les résultats obtenus en un temps réduit.

V Bibliographie

[1] HANNACHE, Aboubakr et LEMMOUIS, Abdelhamid. Résolution d'un problème d'ordonnancement de type job shop avec contrainte de transport. Mémoire master, université Abou Bakr Belkaid. 2016.

[2] F.A. Rodammeret K. Preston White, « A recent survey of production scheduling ». IEEE Transaction on Systems, Man and Cybernetics, pp. 6-18, 1999.

[3] J. Carlier et P. Chrétienne, « Problèmes d'ordonnancement, Modélisation, Complexité, Algorithmes ». Edition Masson, Paris, 1988.

[4] KACIMI EL HASSANI Mounya, étude comparative des algorithmes génétiques pour un problème d'ordonnancement d'atelier M-machines identiques en parallèles, mémoire de fin d'étude Master, université Mohamed Boudiaf-Msila.2019.

[5]SITAYEB, Fatima Benbouzid. Contribution à l'étude de la performance et de la robustesse des ordonnancements conjoints Production/Maintenance-Cas du Flowshop. 2005. Thèse de doctorat.

[6] P. Esquerol et P. Lopez, 1999, « L'ordonnancement », Economica.

[7] B. Roy et D. Bouysson, « Aide multi-critères à la décision : Méthodes et cas ».Collection Gestion Série : Production et technologie quantitatives appliquées à la gestion.EditionEconomica, Paris, 1993.

[8]KARRAY, Asma. Contribution à l'ordonnancement d'ateliers agroalimentaires utilisant des méthodes d'optimisation hybrides. 2011. Thèse de doctorat. Ecole centrale de Lille.

[9] : C. Esswein. « Un apport de flexibilité séquentielle pour l'ordonnancement robuste ». thèse de doctorat, Université François Rabelais, Tours, France, Décembre 2003

[10] : Aziza BECHIR, Résolution de problèmes d'optimisation par les systèmes multi-agents et les approches évolutionnaires, mémoire magister en informatique, université Mohamed khider Biskra. 2016.

[11] Algorithmes de construction de graphes dans les problèmes d'ordonnancement de projet. Mouhoub Nasser Eddine. Mémoire master, université Ferhat Abbas, Sétif, 2014.

[12] Boukef Ben Othman H., « Sur l'ordonnancement d'ateliers job-shop flexibles et flow shop en industries pharmaceutiques Optimisation par algorithmes génétiques et essaims particulaires», Thèse de doctorat, Ecole Centrale De Lille, 2009.

[13] Pinedo M., « Scheduling: Theory, Algorithms and Systems », Third edition, Springer, 2008.

[14] BAPTISTE P. and LE PAPE C., « A Constraint Based Branch and Bound Algorithm for Preemptive Job Sop Scheduling », Proceedings of the International Workshop on Production Planning and Control, Mons, Belgium, 1996.

[15] Bronson R., Naadimuthu G., « Operation research », Second edition, Schaum's Outline Séries, 1997.

[16] Y. Collette et P. Siarry, « Optimisation Multiobjectif ». Editions Eyrolles, Paris, 2002.

[17] R.E. Bellman, « The Bellman continuum ». Editions Robert S. Roth, 1986.

[18] Nadir Ramla, mémoire de fin d'étude, Un problème d'ordonnancement de type Job Shop dans un environnement dynamique. Université Mohammed Boudiaf, Msila, 2019.

[19]. BERKANI Abdelhakim, Métaheuristique Hybride Réseaux de Neurones Artificiels-PSO du Recuit Simulé pour la Commande d'un Procédé Industriel Non-linéaire, Mémoire magistère. Université de Batna, 2013.

[20] Alaoui Abdiya, Application des techniques des métaheuristiques pour l'optimisation de la tâche de la classification de la fouille de données. Thèse magistère, université d'Oran, 2012.

[21]. Sébastien, CAHON. ParadisEO : Une plate-forme pour la conception et le déploiement de métaheuristiques parallèles hybrides sur clusters et grilles. Thèse de doctorat, 2005.

[22] Deffas Zineb, Métaheuristiques parallèles à solution unique pour la résolution du problème du Q3AP sur grille de calcul. Mémoire magister. université Oum El Bouaghui, 2010.

[23] BEN AHMED Razika, KHERROUBI Zakiya, Optimisation d'un problème d'ordonnancement de type job shop avec contrainte de transport. Mémoire master, université Abou Bakr Belkaid, 2017.

[24] Baptiste Mille, Méthodes approchées pour la résolution d'un problème d'ordonnancement avec travaux interférants. Rapport final de PFE, université de Tours, 2014.

[25] Belkaid Fayçal, cours ordonnancement, introduction à la modélisation (Application sur les problèmes d'ordonnancement à une seule machine).

[26] Selougha Samir, Halassa Zakaria Utilisation de l'heuristique Shifting Bottleneck pour résoudre le problème d'ordonnancement de type Job Shop. Mémoire master, université Abou Bakr Belkaid.2019.

[27] G. Cheikh, J. Boukachour, A. Alaoui, and F. El Khoukhi, Méthode hybride pour l'ordonnancement d'atterrissage d'avions basé sur une modélisation job shop, Université de Rabat, Maroc, 2007.

[28] Younes BAHMANI, Optimisation multicritère de l'ordonnancement des activités de la production et de la maintenance intégrées dans un atelier Job Shop. Thèse de doctorat en 2017, université de Batna.

[29] M. Ennigrou and K. Ghédira, Approche multi-agents basée sur la recherche tabou pour le job shop flexible, IPEIM et ENSI,Tunisie, 2000.

[30] Houari Habiba, Mémoire de Magister en Automatique, Planification et Ordonnancement en temps réel d'un Job shop en utilisant l'Intelligence Artificielle.2012.

[31] Carlier J., et Chretienne, P., Problèmes d'ordonnancement - modélisation - complexité - algorithmes, ERI Masson, 1988.

[32] Mebarki N., Une approche d'ordonnancement temps réel basée sur les règles de priorité des files d'attente. Thèse de doctorat soutenue à l'Université de Claude Bernard Lyon1, 1995.

[33] K.Mebarek, Utilisation des stratégies Méta-heuristiques pour l'ordonnancement des ateliers de type Job Shop, Magister, BATNA, 2008.

[34] P. Lopez et P. Esquirol, «L'ordonnancement,» Economica, Paris, 1999.

[35] J. Carlier, P. Chretienne, (1988). Les Problèmes d'ordonnancement. Masson, Paris, France.

[36] Deddouche Yamina, contribution à l'ordonnancement réactif : modélisation booléenne, mémoire magister en 2012, université d'Oran.

[37] Mohand Larabi. Le problème de job-shop avec transport : modélisation et optimisation. Autre [cs.OH]. Université Blaise Pascal - Clermont-Ferrand II, 2010. Français. NNT: 2010CLF22092. tel-00625528.

[38] P. Lopez et P. Esquirol, «L'ordonnancement,» Economica, Paris, 1999.

[39] HANNACHE, Aboubakr et LEMMOUIS, Abdelhamid. Résolution d'un problème d'ordonnancement de type job shop avec contrainte de transport, mémoire master, université Abou Bakr Belkaid. 2016.

[40] Anthony Caumond. Le problème de jobshop avec contraintes: modélisation et optimisation. Algorithme et structure de données [cs.DS]. Université Blaise Pascal - Clermont-Ferrand II, 2006. Français.

[41] Anthony Caumond. Le problème de jobshop avec contraintes: modélisation et optimisation. Algorithme et structure de données [cs.DS]. Université Blaise Pascal - Clermont-Ferrand II, 2006. Français.

[42] DRISS IMEN, Analyse D'un Système Job Shop Aspect Ordonnancement, Thèse de Doctorat en 2016, Université de Batna 2.

[43] Youssef Harrath. Contribution à l'ordonnancement conjoint de la production et de la maintenance : Application au cas d'un Job Shop.. Automatique / Robotique. Université de Franche-Comté, 2003.

[44] Wajdi Trabelsi, ordonnancement des systèmes de production flexibles soumis à différents types de contraintes de blocage, thèse université de Lorraine.

[45] Wajdi Trabelsi, ordonnancement des systèmes de production flexibles soumis à différents types de contraintes de blocage, thèse université de Lorraine.

[46] Selougha Samir, Halassa Zakaria Utilisation de l'heuristique Shifting Bottleneck pour résoudre le problème d'ordonnancement de type Job Shop, mémoire master en 2019, université Abou Bakr Belkaid.

[47]. https://www.next-decision.fr/editeurs/pmo/ms-project.

[48] http://web-static.stern.nyu.edu/om/software/lekin/.

Les exemples sont générés aléatoirement sous-programme matlab.

Exemples 3 jobs 4 machines

Les exemples	La durée	La gamme
1	Durée= = O1 O2 O3 O4 J1 24 27 8 28 J2 27 18 16 4 J3 3 2 28 29	Gamme = M1 M2 M3 M4 14 12 13 11 21 23 22 24 32 31 33 34
2	Durée= O1 O2 O3 O4 J1 1 20 1 22 J2 2 9 13 23 J3 24 28 11 5	Gamme = M1 M2 M3 M4 12 11 13 14 22 24 23 21 32 31 33 34
3	Durée= O1 O2 O3 O4 J1 28 4 7 27 J2 16 7 24 10 J3 4 25 7 5	Gamme = M1 M2 M3 M4 11 14 13 12 23 22 21 24 31 34 33 32
4	Durée= O1 O2 O3 O4 J1 14 4 15 17 J2 23 4 19 0 J3 10 9 18 20	Gamme = M1 M2 M3 M4 13 14 12 11 0 22 23 21 33 34 31 32
5	Durée= O1 O2 O3 O4 J1 24 5 4 16 J2 12 7 26 4 J3 27 4 17 25	Gamme = M1 M2 M3 M4 12 14 13 11 21 23 24 22 33 31 32 34

Exemples 4 jobs 4 machines

Les exemples	La durée	La gamme
1	Durée= = O1 O2 O3 O4 J1 24 18 28 28 J2 27 2 28 14 J3 3 8 4 24 J4 27 16 29 4	Gamme = M1 M2 M3 M4 11 13 12 14 22 21 23 24 34 31 33 32 44 42 41 43
2	Durée= O1 O2 O3 O4 J1 26 24 7 17 J2 2 12 4 16 J3 11 27 4 4 J4 7 5 26 25	Gamme = M1 M2 M3 M4 12 14 13 11 21 23 24 22 33 31 32 34 44 43 42 41
3	Durée= O1 O2 O3 O4 J1 27 11 15 17 J2 8 17 23 14 J3 22 2 28 0 J4 22 1 3 10	Gamme = M1 M2 M3 M4 11 13 14 12 21 23 22 24 33 31 32 0 41 44 42 43
4	Durée= O1 O2 O3 O4 J1 28 10 5 13 J2 17 24 19 16 J3 1 21 8 0 J4 7 1 19 22	Gamme = M1 M2 M3 M4 13 11 14 12 23 21 22 24 33 0 32 31 42 41 43 44
5	Durée= O1 O2 O3 O4 J1 18 14 6 9 J2 17 6 5 27 J3 6 25 6 12 J4 9 5 13 5	Gamme = M1 M2 M3 M4 14 13 11 12 21 24 22 23 34 33 31 32 41 42 43 44

Exemples 8 jobs 4 machines

Les exemples	La durée	La gamme
1	Durée= = O1 O2 O3 O4 J1 7 24 17 14 J2 27 17 2 0 J3 10 16 1 10 J4 5 27 15 4 J5 7 8 23 23 J6 18 22 28 9 J7 14 22 3 15 J8 10 11 17 4	Gamme = M1 M2 M3 M4 12 11 13 14 23 22 21 0 33 31 34 32 43 44 41 42 53 51 54 52 63 64 61 62 71 73 72 74 84 83 81 82
2	Durée= O1 O2 O3 O4 J1 18 19 22 8 J2 28 21 1 29 J3 11 11 6 4 J4 12 18 24 14 J5 1 19 4 18 J6 23 9 10 16 J7 12 29 9 13 J8 20 17 12 0	Gamme = M1 M2 M3 M4 11 13 14 12 21 23 22 24 31 34 33 32 43 44 42 41 52 54 53 51 61 63 64 62 71 72 73 74 83 81 0 82
3	Durée= O1 O2 O3 O4 J1 3 5 20 6 J2 6 16 27 21 J3 20 3 2 29 J4 10 11 24 26 J5 9 18 14 27 J6 6 7 9 6 J7 20 16 21 16	Gamme = M1 M2 M3 M4 12 13 11 14 22 23 24 21 34 31 32 33 42 41 43 44 53 51 54 52 61 64 63 62 73 72 74 71

	J8 26 23 7 9	83 81 84 82
4	Durée= O1 O2 O3 O4 J1 11 11 5 22 J2 4 11 26 18 J3 2 4 13 9 J4 27 7 18 15 J5 14 3 13 27 J6 13 17 15 23 J7 9 5 24 10 J8 11 12 2 24	Gamme = M1 M2 M3 M4 11 14 13 12 24 21 22 23 32 31 33 34 43 42 44 41 51 52 53 54 63 61 62 64 73 71 74 72 82 81 83 84
5	Durée= O1 O2 O3 O4 J1 2 8 6 11 J2 5 28 6 9 J3 25 4 18 0 J4 23 25 18 19 J5 21 12 4 20 J6 5 27 19 16 J7 7 8 29 1 J8 8 13 24 26	Gamme = M1 M2 M3 M4 12 14 11 13 22 21 23 24 33 0 31 32 43 42 44 41 51 53 54 52 62 64 61 63 71 74 73 72 82 83 81 84

Exemples 10 jobs 4 machines

Les exemples	La durée	La gamme
1	Durée= = O1 O2 O3 O4 J1 16 3 13 4 J2 23 2 24 7 J3 21 12 20 28 J4 4 2 19 8 J5 8 12 18 13 J6 25 21 14 25 J7 15 13 2 20 J8 16 1 2 17 J9 11 20 25 3 J10 5 17 6 24	Gamme = M1 M2 M3 M4 12 14 11 13 23 21 24 22 32 34 33 31 41 42 43 44 52 51 53 54 62 63 61 64 71 74 73 72 82 84 81 83 93 92 94 91 101 103 104 102
2	Durée= O1 O2 O3 O4 J1 11 10 2 27 J2 10 2 9 21 J3 21 22 27 21 J4 22 6 29 5 J5 11 10 14 27 J6 14 23 3 11 J7 20 10 18 0 J8 3 16 7 20 J9 25 12 4 19 J10 29 10 15 20	Gamme = M1 M2 M3 M4 11 13 14 12 21 24 22 23 31 34 33 32 41 42 43 44 54 53 52 51 61 62 64 63 71 72 73 0 81 84 83 82 94 92 91 93 101 102 103 104
3	Durée= O1 O2 O3 O4 J1 5 18 11 27 J2 13 6 8 4 J3 28 29 4 12	Gamme = M1 M2 M3 M4 11 14 13 12 21 24 22 23 33 32 34 31

	J4 9 15 17 0 J5 11 7 26 4 J6 15 11 5 24 J7 4 15 11 8 J8 18 27 12 8 J9 8 28 18 3 J10 19 13 20 12	42 0 43 41 51 52 54 53 64 63 62 61 74 72 73 71 81 84 82 83 91 92 93 94 103 102 104 101
4	Durée= O1 O2 O3 O4 J1 11 4 19 24 J2 12 19 8 2 J3 2 12 19 24 J4 7 15 10 8 J5 14 10 13 0 J6 15 3 11 13 J7 21 28 13 14 J8 1 19 11 25 J9 3 4 16 13 J10 29 25 5 0	Gamme = M1 M2 M3 M4 11 14 13 12 23 24 22 21 32 34 33 31 43 41 44 42 0 53 51 52 64 61 62 63 73 71 74 72 81 83 82 84 93 94 91 92 102 101 0 103
5	Durée= O1 O2 O3 O4 J1 14 8 22 5 J2 8 13 29 3 J3 28 10 22 17 J4 26 28 24 24 J5 13 29 15 11 J6 3 15 27 23 J7 18 7 3 1 J8 4 15 26 23 J9 14 16 9 25 J10 7 4 9 23	Gamme = M1 M2 M3 M4 13 12 11 14 22 21 24 23 32 33 31 34 43 44 41 42 54 53 51 52 64 62 61 63 73 74 72 71 84 83 81 82 91 94 93 92 102 103 101 104

Exemples 20 jobs 4 machines

Les exemples	La durée	La gamme
1	Durée= O1 O2 O3 O4 J1 29 26 16 5 J2 12 13 0 23 J3 5 13 15 16 J4 12 29 16 20 J5 8 25 20 5 J6 11 4 23 25 J7 19 28 3 0 J8 20 14 25 28 J9 0 5 1 13 J10 4 8 19 9 J11 3 14 8 0 J12 20 7 16 18 J13 3 3 2 18 J14 13 17 23 15 J15 25 8 20 21 J16 24 25 4 25 J17 3 23 11 0 J18 19 16 8 17 J19 28 19 27 24 J20 1 5 17 7	Gamme = M1 M2 M3 M4 13 14 11 12 23 21 0 22 32 33 31 34 41 44 43 42 52 53 51 54 61 64 62 63 71 73 0 72 84 83 81 82 0 93 91 92 104 101 102 103 112 113 111 0 124 122 121 123 131 133 132 134 142 143 144 141 154 152 151 153 164 162 163 161 171 172 0 173 181 183 184 182 191 194 192 193 201 204 202 203
2	Durée= O1 O2 O3 O4 J1 16 4 26 3 J2 29 8 28 20 J3 29 8 27 3	Gamme = M1 M2 M3 M4 13 12 14 11 21 22 23 24 34 31 33 32

	J4	15	1	11	28	42	43	41	44
	J5	22	11	7	24	54	51	52	53
	J6	12	12	21	24	64	63	61	62
	J7	9	22	21	17	74	72	73	71
	J8	7	23	27	12	82	81	84	83
	J9	9	29	15	6	93	94	91	92
	J10	21	12	24	1	102	104	103	101
	J11	18	15	0	29	112	113	0	111
	J12	24	21	27	22	121	122	123	124
	J13	28	5	11	8	133	131	134	132
	J14	27	7	3	20	143	141	142	144
	J15	15	10	10	27	153	154	152	151
	J16	14	16	1	27	164	163	161	162
	J17	8	5	5	22	173	172	174	171
	J18	3	7	17	5	183	182	181	184
	J19	27	23	24	29	191	194	192	193
	J20	22	16	3	0	0	201	203	202
3	Durée=	O1	O2	O3	O4	Gamme = M1	M2	M3	M4
	J1	16	26	22	10	11	14	13	12
	J2	26	17	25	8	24	21	22	23
	J3	28	27	12	20	31	32	34	33
	J4	9	8	7	26	44	42	41	43
	J5	10	20	7	27	52	53	51	54
	J6	8	19	1	6	61	64	63	62
	J7	16	24	27	1	73	74	71	72
	J8	29	22	29	21	83	84	81	82
	J9	15	19	0	10	93	92	0	91
	J10	19	10	2	26	101	102	104	103
	J11	23	24	4	17	113	112	114	111
	J12	14	2	26	21	123	124	121	122
	J13	21	22	21	24	131	133	134	132

	J14	22	10	23	3	143	144	142	141
	J15	1	23	20	26	152	153	151	154
	J16	11	21	6	18	162	163	164	161
	J17	3	12	29	28	173	172	171	174
	J18	27	20	27	14	181	184	182	183
	J19	15	19	19	29	191	194	192	193
	J20	19	21	24	16	202	203	204	201
4	Durée=	O1	O2	O3	O4	Gamme = M1	M2	M3	M4
	J1	28	12	6	14	12	14	11	13
	J2	15	25	18	28	24	22	21	23
	J3	22	23	29	16	34	33	31	32
	J4	11	28	15	3	44	41	42	43
	J5	6	23	13	0	53	0	52	51
	J6	16	7	19	23	62	63	64	61
	J7	14	25	20	4	71	74	73	72
	J8	0	15	6	9	0	83	81	82
	J9	4	7	27	7	91	92	94	93
	J10	7	28	14	11	102	103	101	104
	J11	17	27	24	1	112	114	111	113
	J12	12	16	4	15	123	121	122	124
	J13	23	6	6	22	132	131	134	133
	J14	18	6	17	27	141	144	143	142
	J15	17	5	13	1	153	151	154	152
	J16	4	26	5	11	163	164	162	161
	J17	13	9	8	6	172	174	171	173
	J18	9	17	13	25	183	184	182	181
	J19	1	24	16	17	192	191	194	193
	J20	26	16	9	11	204	201	202	203
	Durée=	O1	O2	O3	O4	Gamme = M1	M2	M3	M4
	J1	11	1	6	17	13	12	11	14
	J2	7	17	19	0	21	23	22	0

5	J3	11	24	18	10	33	32	34	31
	J4	5	8	9	6	43	42	44	41
	J5	29	10	22	0	53	0	52	51
	J6	25	8	21	18	61	64	62	63
	J7	22	19	20	5	71	74	73	72
	J8	10	24	7	3	83	81	82	84
	J9	5	7	29	12	92	94	91	93
	J10	11	13	24	23	101	103	102	104
	J11	5	22	13	9	114	111	113	112
	J12	15	21	27	4	124	122	121	123
	J13	10	23	8	8	131	132	134	133
	J14	21	8	16	2	144	142	143	141
	J15	15	9	17	24	154	151	152	153
	J16	11	14	3	3	163	162	164	161
	J17	5	26	22	15	172	171	174	173
	J18	10	23	20	17	181	182	183	184
	J19	22	20	19	0	0	193	192	191
	J20	1	22	13	14	202	201	203	204

الملخص :غالبًا ما يتم تصنيف مشكلات الجدولة لنظام إنتاج نوع جوب شوب على أنها صعبة جدا . يتطلب حل مثل هذه المشكلات طرقًا مخصصة ، في حين أن الأساليب الدقيقة لا يمكنها حل هذه الأنواع من المشكلات بسبب الوقت الحسابي الضخم ، فإن الاستدلال يوفر إمكانية إيجاد حل عملي في وقت معقول .درسنا في هذا العمل أداء ثلاثة أساليب استكشافية مستخدمة لحل هذا النوع من المشاكل ، وهي الوقت القصير,الوقت الكبير وتحويل استدلال عنق الزجاجة . من خلال النتائج المتحصل عليها لاحظناها أن عنق الزجاجة الإرشادي قدم نتائج مرضية لمشاكل الحجم الصغير ، وللمشكلات المتوسطة وكبيرة الحجم أعطى الوقت الكبير أفضل النتائج

لكلمات المفتاحية :جدولة جوب شوب، الاستدلال ، المدى ، قاعدة التفضيل

Résumé : Les problèmes d'ordonnancement d'un système de production de type job shop sont souvent classés NP-Difficiles. La résolution de tels problèmes nécessite des méthodes dédiées, tandis que les méthodes exactes ne peuvent pas résoudre ces types de problèmes vu le temps de calcul énorme, les heuristiques offrent la possibilité de trouver une solution réalisable en un temps raisonnable. Dans ce travail nous avons étudié la performance de trois heuristiques utilisées pour la résolution de ce type de problèmes à savoir l'heuristique SPT, LPT et shifting bottleneck. D'après les résultats obtenus nous avons remarqué que l'heuristique shifting bottleneck fourni des résultats satisfaisants pour les problèmes de petites tailles, et pour les problèmes de moyennes et grandes tailles l'heuristique LPT (avec disponibilité de machines) donne les meilleurs résultats.

Mots clés: Ordonnancement Job Shop, heuristiques, Makespan, Règle de priorité.

Abstract: The scheduling problems of a job shop type production system are often rated NP-Difficult. Solving such problems requires dedicated methods, while exact methods cannot solve these types of problems due to the huge computational time, heuristics offer the possibility of finding a workable solution in a reasonable time. In this work we studied the performance of three heuristics used for solving this type of problem, namely SPT, LPT and shifting bottleneck heuristics. From the results we have observed that the heuristic shifting bottleneck provided satisfactory results for small size problems, and for medium and large size problems the LPT heuristic (with machine availability) gave the best results.

Keywords: Job shop Scheduling, heuristics, Makespan, Dispatching rules.