Table des matières

Introduction

La concurrence industrielle, nécessite d'utiliser au mieux toutes les ressources potentielles de l'entreprise. Cette optimisation doit traiter les problèmes plus spécifiques à tous les niveaux, et notamment au niveau de la production et la gestion de ressources.

Dans ce projet, on va s'intéresser d'une part, aux problèmes d'ordonnancement et particulièrement aux problèmes d'ordonnancement d'atelier. Ainsi, l'étude porte sur l'ordonnancement d'un ensemble de jobs sur un ensemble de ressources dans divers environnements (Job shop, Flow shop……etc.).
D'autre part sur les différents types de modélisation et de résolution d'un problème d'ordonnancement d'atelier de type Job shop et Flow shop.
Le dernier chapitre concerne les méthodes d'optimisation des problèmes d'ordonnancement

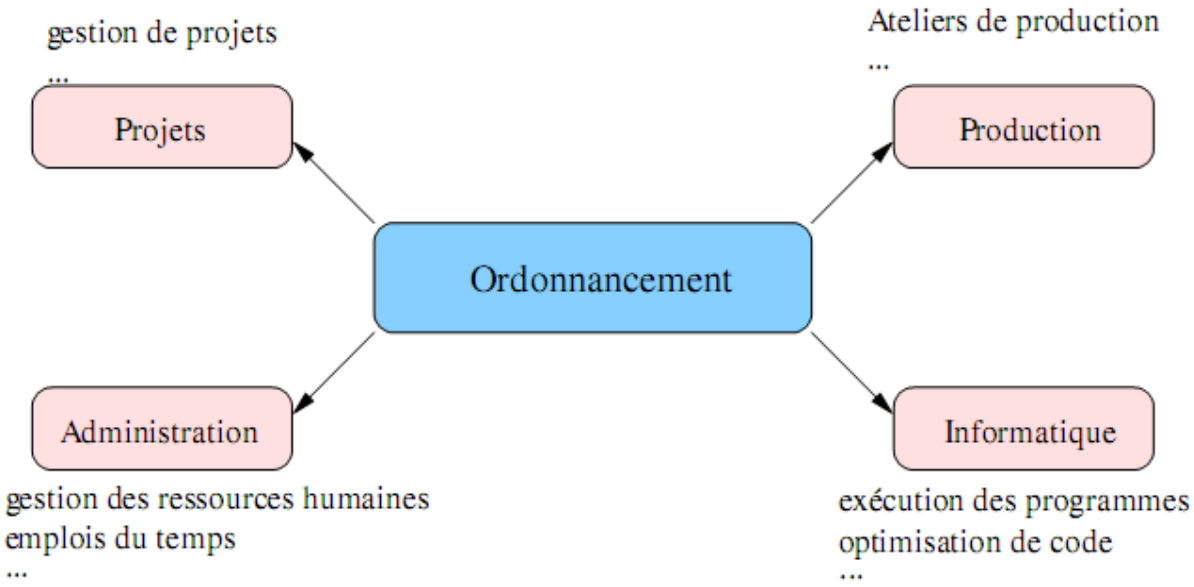
I- problème d'ordonnancement

1-définitions et notions fondamentales :

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» [8].

Les domaines concernés [7] ***:***



Dans un problème d'ordonnancement interviennent deux notions fondamentales : les ressources et les tâche.

- ✓ Une ressource est un moyen technique ou humain dont la disponibilité limitée ou non est connu à priori [1].

- ✓ Une tâche est un travail élémentaire dont la réalisation nécessite un certain nombre d'unités de temps (sa durée) et d'unités de chaque ressource [1], 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].

2- Notion d'objectifs et de critères :

Les objectifs des entreprises sont diversifiés et le processus d'ordonnancement est devenu de plus en plus multicritère. Les critères que doivent satisfaire un ordonnancement sont variés.

D'une manière générale, on distingue plusieurs classes d'objectifs concernant un ordonnancement [4]:

§ Les objectifs liés au temps : on trouve par exemple la minimisation du temps total d'exécution(C_{max}), du temps moyen d'achèvement, des durées totales de réglage ou des retards par rapports aux dates de livraisons (L_{max} ou T_{max})

§ Les objectifs liés aux ressources : maximiser la charge d'une ressource ou minimiser le nombre de ressources nécessaire pour réaliser un ensemble des tâches.

§ Les objectifs liés au coût : minimiser les coûts de lancement, de production, de stockage, de transport, etc.

§ Les objectifs liés à une énergie ou un débit

3- contraintes d'ordonnancement :

Une contrainte exprime des restrictions sur les valeurs que peuvent prendre simultanément les variables représentant les relations reliant les tâches et les ressources. On distingue deux types de contraintes, **les contraintes temporelles et les contraintes de ressources.**

- **Les contraintes temporelles :**

Les contraintes temporelles concernent les délais de fabrication imposés. Ces contraintes peuvent être [4]:

- **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**.**
- **des contraintes de dates au plus tôt,** liées à l'indisponibilité de certains facteurs nécessaires pour commencer l'exécution des tâches.

- **Les contraintes de ressources *:***

Ces contraintes concernent la limitation de la quantité des ressources de chaque type. Dans ce cadre, deux types de contraintes de ressources sont distingués [4]:

- **Les contraintes disjonctives** : induisant une contrainte de réalisation des tâches sur des intervalles temporels disjoints pour une même ressource.
- **Les contraintes cumulatives** : impliquant la limitation du nombre de tâches à réaliser en parallèle.

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 :

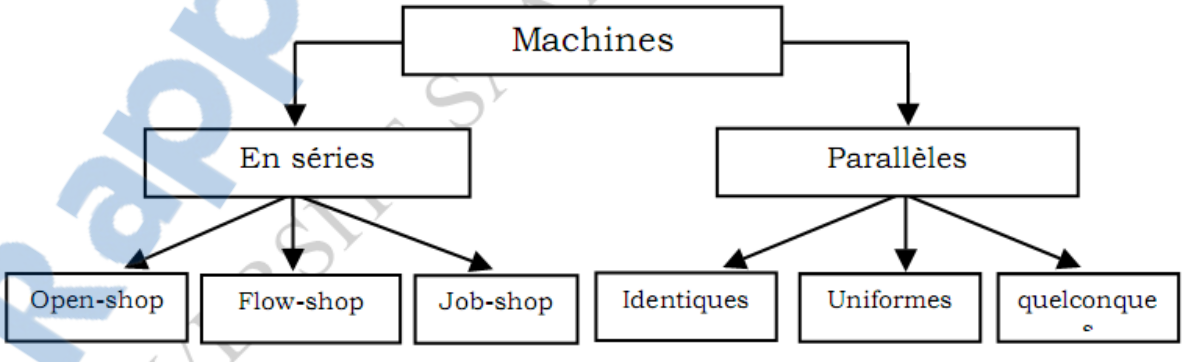
- ✓ L'aspect statique consiste à générer un plan de réalisation des travaux sur la base des données prévisionnelles.
- ✓ L'aspect dynamique consiste à prendre des décisions en temps réel, compte tenu de l'état des ressources et l'avancement dans le temps des différentes taches.

II- Les problèmes d'ordonnancement d'atelier

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 n_i opérations, l'opération j du travail i est notée (i,j) avec (i, 1) < (i, 2) < ... < (i, n_i) si le travail i possède une gamme (A < B signifie A précède B). Une opération (i,j) utilise la machine $m_{i,j}$ pendant toute sa durée $p_{i,j}$ 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).

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



1. Schémas de classification [7] [8]*:*

Nous suivons les schémas de classification proposée par (Graham et al, 1979).

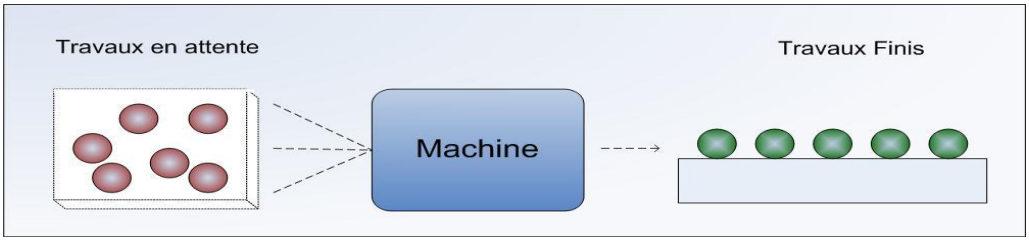
Classification à trois champs **α, β, γ** :

- **α : environnement machine.**
  - **α_1 :** Prend des différents caractères selon le type de problème**.**
  - **α_2 :** Désigne le nombre de machine .Si α2 est entier positive, le nombre de machine supposé constant .si α2 est absent alors ce nombre est supposé arbitraire.
- **β : les caractéristiques des tâches.**
  - **β_1 =** pmtn si la préemption des tâches est autorisée, sinon β1 est absent.
  - S'il y a des contraintes de précédence entre les tâches **β_2** $\in$ {prec, chain, tree }, sinon **β_2** est vide.
  - **β_3 =** r_j si les dates de début au plus tôt r_j (ou dates de disponibilité) des tâches ne sont pas forcément identiques, sinon ($\forall$j, $r_j = 0$) β3 est absent.
  - **β_4 =** $\tilde{d}_j$ si la tâche ou le job possède une date de fin obligatoire.
- **γ : le (ou les) critère(s) à optimiser.**

2. Les types des problèmes :

2.1. Problèmes à une machine :

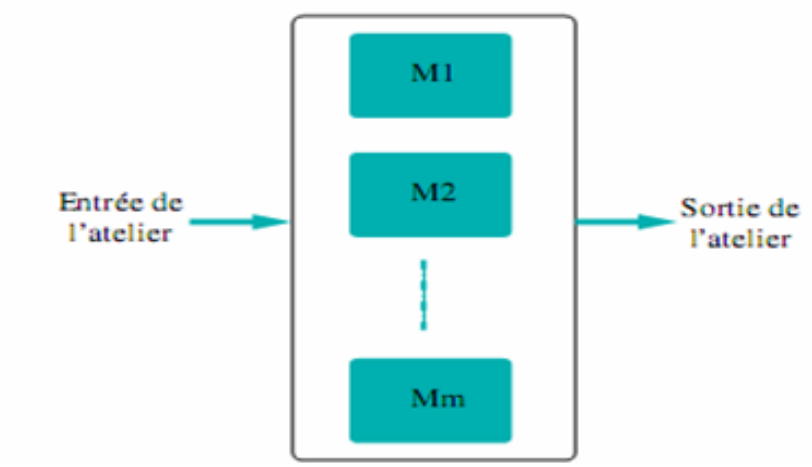
Toute tâche J_j ,j$\in$ {1,….,n} de durée P_j (processing time) s'exécute sur une machine qui ne peut traiter plus qu'une tache à la fois [5] [6] [7] [8].Le champ α_1 est absent et $\alpha_2 = 1$.



Figur 2.1

2.2. Problèmes à machines parallèles :

Toute tâche J_j ,$j \in \{1,\ldots,n\}$ peut être exécutée indifféremment sur une des m machines mises en parallèle [5] [6] [7] [8].



Figur 2.2

- Si toutes les machines ont la même vitesse d'exécution des tâches, les machines sont appelées « identiques » et le problème noté (P).
- Si les machines sont différentes par leur vitesse d'exécution et la vitesse de chaque machine est constante et ne dépend pad de l'ensemble des tâches, elles sont dites « uniformes ». (Q).
- Si les vitesses d'exécution des machines dépendent des tâches et sont différentes alors elles sont dites « quelconques » ou différentes (R).

2.3. Problèmes de Flow shop :

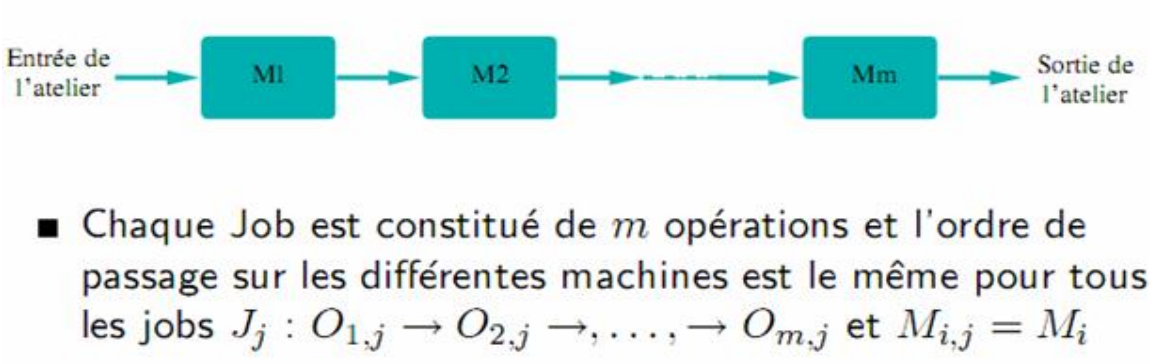
Définition : On appelle gamme d'un travail ou gamme opératoire, la succession des opérations qui le constituent, ou la succession des machines qui sont associées. Les contraintes et les critères du problème concernent les opérations, leurs localisations temporelles, et les moyens nécessaires à leur réalisation.

• **Les problèmes de Flow shop :** la gamme de fabrication, qui correspond à l'ordre d'exécution des opérations au sein des travaux, est linéaire, fixée à l'avance et identique pour tous les travaux, ce qui permet de numéroter les machines dans l'ordre d'exécution des opérations à traiter [5] [6] [7] [8]. Selon les types de produits élaborés, on distingue **la production continue** et **la production discrète.** La production continue est caractérisée par

la fluidité de son processus et l'élimination du stockage. C'est le cas notamment dans les raffineries, les cimenteries, les papeteries... La production discrète de masse s'applique principalement aux produits de grande consommation fabriqués à la chaîne (e.g automobile, la majorité du domaine du textile, machines-outils…).

Parmi les caractéristiques d'un problème de cette catégorie :

- il existe au minimum n! différentes solutions où n est le nombre de travaux à réaliser. Notons que n! = n*(n-1)*(n-2)….*1 ;
- une grande productivité mais une faible flexibilité. (voir la notion de la flexibilité en bas)



Figur 2.3

- **α_1=F**

2.4. Problèmes de job shop :

• **Les problèmes de job shop** : la gamme de fabrication est linéaire et fixée à l'avance, mais peut varier d'un travail à l'autre. Une tâche peut revenir une seconde fois sur la même machine [5] [6] [7] [8]. L'une des caractéristiques d'un atelier à cheminement multiple est que la demande pour un produit particulier est généralement d'un volume petit ou moyen. Une autre caractéristique est la variabilité dans les opérations et un mix produit constamment changeant.

-α_1=J

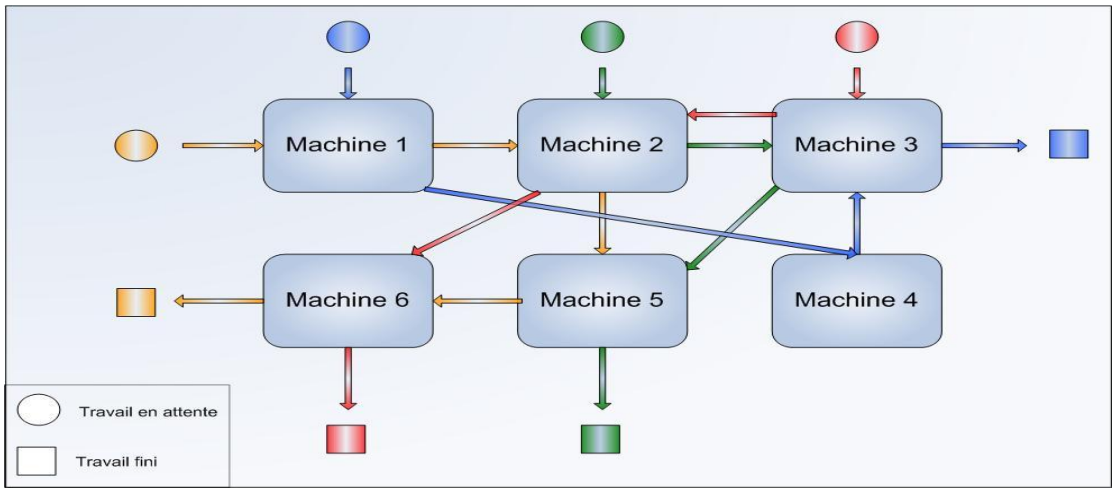


Figure2.4

- Le nombre d'operation n'est pas forcément le meme pour tous les jobs.
- Chaque job a son propre ordre de passage sur les machines.

Parmi les autres caractéristiques d'un problème d'ordonnancement dans un atelier à cheminements multiples :

- le nombre de solutions possibles est de l'ordre de (n!)m, où n est le nombre de tâches à effectuer et m le nombre de machines. Notons qu'une tâche veut dire la même chose qu'un travail.
- le problème est considéré parmi les problèmes les plus difficiles à traiter.

2.5. Problèmes d'open shop :

• **Les problèmes de open shop** : C'est un modèle d'atelier moins contraint que le Flow shop et le job shop, car l'ordre des opérations n'est pas fixe à priori [5] [6] [8]. Le problème d'ordonnancement consiste d'une part à déterminer le cheminement de chaque travail et d'autre part à ordonnancer les travaux en tenant compte des gammes trouvées (Un problème de type open shop est un job shop dans lequel les contraintes de précédence sont relâchées. C'est-à dire, les opérations peuvent être effectuées dans n'importe quel ordre).

3. La notion de la flexibilité :

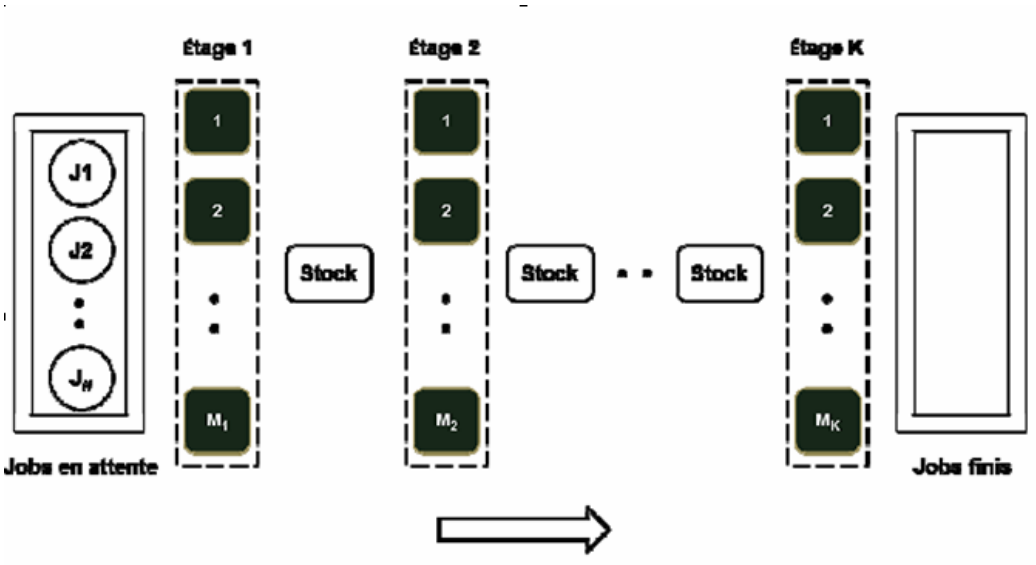
La **flexibilité** est une mode de gestion de la main d'œuvre qui permet aux entreprises d'adapter rapidement la production et l'emploi (l'offre) aux fluctuations rapides des commandes des clients (demande), et il y a cinq types de flexibilité du travail :

- **la flexibilité externe quantitative** qui permet de faire fluctuer les effectifs de l'entreprise en fonction des besoins en ayant recours aux licenciements et aux contrats de travail de courte durée.
- **la flexibilité externe qualitative (ou externalisation)** qui « consiste à déplacer sur une autre entreprise le lien contractuel avec le travailleur » en aillant recours par exemple aux travailleurs intérimaires ou à l'externalisation d'un certain nombre d'activités annexes à la production (gardiennage, restauration, nettoyage…).
- **la flexibilité salariale** qui permet de faire varier à travers la rémunération des salariés, le coût de la masse salariale de l'entreprise. Elle « est conçue comme un moyen de répercuter sur les salaires les évolutions du chiffre d'affaire et de coûts de revient de l'entreprise en fonction des mouvements conjoncturels ».
- **la flexibilité interne quantitative** qui consiste à faire varier la quantité d'heures travaillées pour un effectif donné. Elle peut être réalisée par des modulations saisonnières à partir d'un contrat portant sur une durée annuelle, des temps partiels, des travails intermittents, des heures supplémentaires...
- **La flexibilité interne qualitative (ou flexibilité fonctionnelle)** qui « consiste, à quantité de travail donnée, à employer les travailleurs à des fonctions variables en fonction des besoins de la chaîne de production ou des fluctuations de la production ».

Afin d'être toujours plus réactives et productives, les entreprises ont cherché à augmenter la flexibilité de leurs systèmes de production. Pour atteindre ce but, il est possible de multiplier le nombre des machines qui peuvent réaliser une même opération. Ces machines, considérées comme identiques dans le cadre de cette thèse, sont regroupées en étage ou cellule. Et pour cela on peut distinguer autres types d'atelier :

3.1. Flow shop hybride :

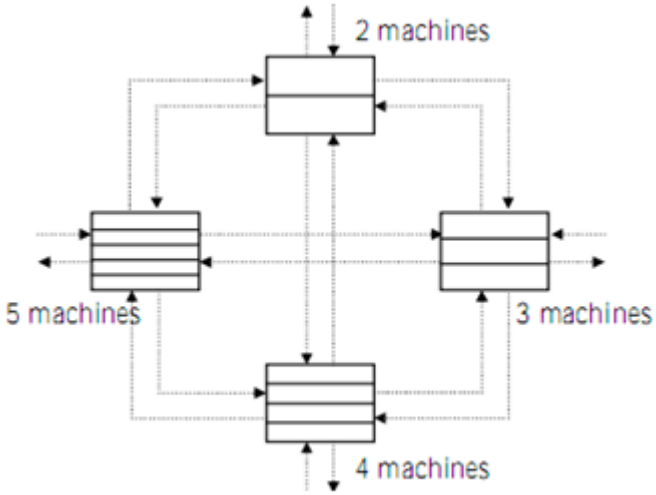
Le Flow Shop hybride est une généralisation du Flow Shop classique au cas où plusieurs machines sont disponibles sur un ou plusieurs étages pour exécuter les différentes tâches du Flow Shop. Ces problèmes présentent alors une difficulté supplémentaire par rapport aux problèmes sans flexibilité des ressources. En effet, la machine qui sera utilisée pour exécuter une opération n'est pas connue d'avance, mais doit être sélectionnée parmi un ensemble donné pour construire une solution au problème [9].



Figur 2.5

3.2. Job shop flexible :

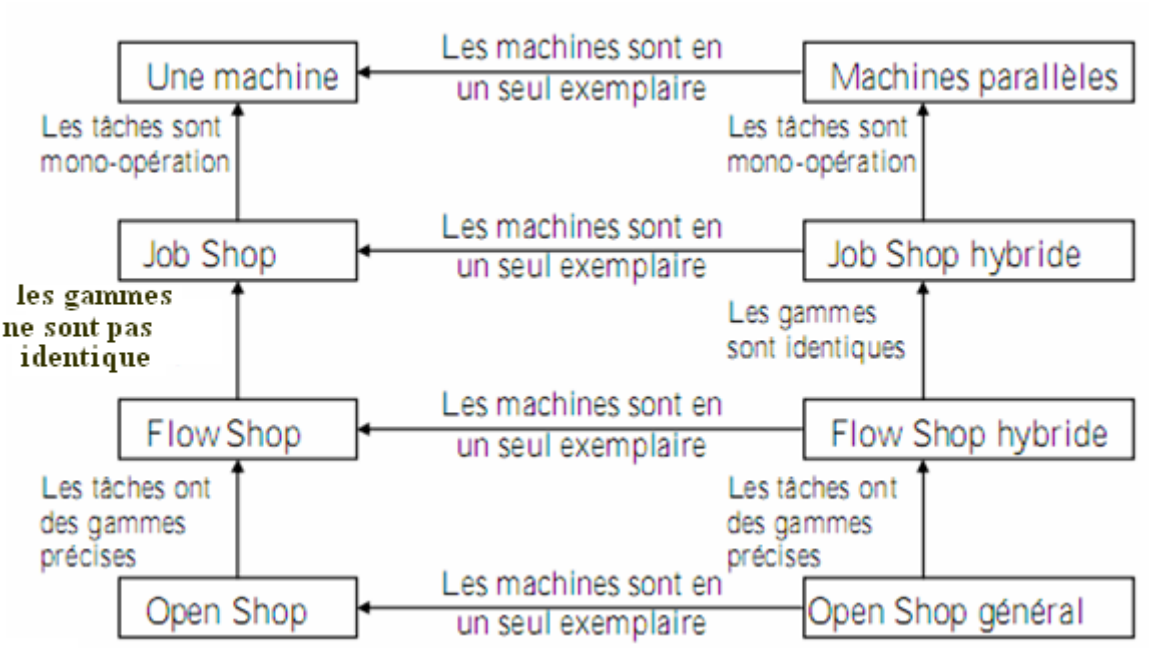
Le job shop flexible est une extension du modèle job shop classique. Sa particularité essentielle réside dans le fait que plusieurs machines sont potentiellement capables de réaliser un sous ensemble d'opérations. Plus précisément, une opération est associée à un ensemble contenant toutes les machines pouvant effectuer cette opération [6].



Figur 2.6

4. Conclusion :

Ce qui concerne les différentes organisations des machines on peut conclure les relations suivantes :



III. Méthodes d'optimisation des problèmes d'ordonnancement :

Etant donnés un ensemble de tâches et un ensemble de ressources, il est nécessaire de programmer les tâches et d'affecter les ressources de façon à optimiser un ou plusieurs objectifs (un objectif correspondant à un critère de performance), en respectant un ensemble de contraintes. La principale difficulté à laquelle est confronté un décideur, en présence d'un problème d'optimisation est celui du choix d'une méthode efficace capable de produire une solution optimale en un temps de calcul raisonnable. Les différentes méthodes de résolution développées peuvent être classées en deux catégories : les méthodes exactes qui garantissent la complétude de la résolution et les méthodes approchées qui perdent la complétude pour gagner en efficacité.

1. Les méthodes exactes :

On peut définir une méthode exacte comme étant une méthode qui fournit une solution optimale pour un problème d'optimisation.

L'utilisation de ce type de méthodes s'avère particulièrement intéressante dans les cas des problèmes de petites tailles. La méthode par séparation et évaluation (branch and bound) constituent certainement celles qui sont les plus utilisées pour résoudre les problèmes d'optimisation multi objectifs [2]. D'autres méthodes telles que la programmation linéaire ou la programmation dynamique, sont aussi utilisées couramment.

Toutes ces méthodes examinent d'une manière implicite, la totalité de l'espace de recherche pour produire la solution optimale[2].

1.1. La méthode de Branch and Bound :

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 .La démarche de l'algorithme Branch and Bound consiste à :

- Diviser l'espace de recherche en sous espaces,
- Chercher une borne minimale en terme de fonction objectif associée à chaque sous espace de recherche,
- Eliminer les mauvais sous-espaces,
- Reproduire les étapes précédentes jusqu'à l'obtention de l'optimum global.

1.2. *La programmation dynamique :*

Elle se base sur le principe de Bellman [Bellman, 86] : « 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. Cette méthode est destinée à résoudre des problèmes d'optimisation à vocation plus générale que la méthode de séparation et d'évaluation (branch and bound) sans permettre pour autant d'aborder des problèmes de tailles importantes.

1.3. La programmation linéaire :

C'est l'une des techniques classiques de recherche opérationnelle. Elle repose sur la méthode du simplexe et les algorithmes de points intérieurs de Karmarkar.
Elle consiste à minimiser une fonction coût en respectant des contraintes, le critère et les contraintes étant des fonctions linéaires des variables du problème.

2. Les heuristiques :

Les heuristiques sont des méthodes empiriques qui donnent généralement de bons résultats sans pour autant être démontrables. Elles se basent sur des règles simplifiées pour optimiser un ou plusieurs critères. Le principe général de cette catégorie de méthodes est

d'intégrer des stratégies de décision pour construire une solution proche de celle optimale tout en cherchant à avoir un temps de calcul raisonnable [Bel, 01]. Parmi ces stratégies, nous distinguons [2]:

-FIFO (First In First Out) où la première tâche arrivée est la première à être ordonnancée,

-SPT (Shortest Processing Time) où la tâche ayant le temps opératoire le plus court est traitée en premier,

-LPT (Longest Processing Time) où la tâche ayant le temps opératoire le plus important est traitée en premier,

-EDD (Earliest Due Date) où la tâche ayant la date due la plus petite est la plus prioritaire.

3. Les méta-heuristiques:

Malgré l'évolution permanente de l'informatique, il existe toujours, pour un problème polynomial, une taille critique au-dessus de laquelle une énumération, même partielle, des solutions admissibles devient prohibitive. Compte tenu de ces difficultés, la plupart des spécialistes de l'optimisation combinatoire ont orienté leurs recherches vers le développement des méta-heuristiques. Une méta-heuristique est souvent définie comme une procédure exploitant au mieux la structure du problème considéré, dans le but de trouver une solution de qualité raisonnable en un temps de calcul aussi faible que possible.

Les méta-heuristiques sont ainsi des méthodes de recherche générales, dédiées aux problèmes d'optimisation difficile. Elles sont, en général, présentées sous forme de concepts. Les principales méta-heuristiques sont celles basées sur la recherche locale, telles que le recuit simulé et la recherche Tabou, et celles basées sur les algorithmes évolutionnistes telles que les algorithmes génétiques ainsi que les algorithmes basés sur la recherche globale tels que les algorithmes de colonies de fourmis et les algorithmes reposant sur la méthode d'optimisation par essaim de particules

IV-Modélisation et optimisation des problèmes d'ordonnancement d'atelier

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

N.B : Dans cette paragraphes le critère qu'on cherche à optimiser c'est le $\mathbf{C_{max}}$ (la duré totale d'exécution).

1. Job shop classique :

On a vu dans le paragraphe précédant que dans un atelier de Job shop chaque tâche possède son propre mode de passage sur les machines.

1.1. Modélisation :

Nous introduisons les notations suivantes, pour exprimer les données principales d'un problème de job shop :

m : Nombre de machine de l'atelier

M = { M_1,….,M_m } : Ensemble de machines

n : Nombre de job

J = { J_1,….,J_n } : Ensemble de jobs

n_i : Nombre d'opération de job J_i

$O_{i,j}$: j éme opération du job J_i

$u_{i,j}$: Machine requise de l'opération $O_{i,j}$ avec $u_{i,j} \in M$

$P_{i,j}$: La duré d'exécution de l'opération $O_{i,j}$

J_i = { $O_{i,l}$,…, $O_{i,j}$,……, $O_{i\,n_i}$} : séquence technologique de job J_i

Le modèle de H.M.Wagner pour le Job Shop classique, que nous notons par (M_c), se présente comme suit [3]:

$$
M_c \begin{cases}
\sum_{o_{i,j}\,/u_{i,j}=\mathrm{M_k}} X_{i,j}^{(l)} = 1 & \text{pour k} = 1, \ldots\ldots, \text{m} \;;\; \text{l} = 1, \ldots\ldots, \text{m}_\text{k} \quad (1) \\
\sum_{l=1}^{m_k} X_{i,j}^{(l)} = 1 & \text{pour i} = 1, \ldots\ldots, \text{n} \;;\; \text{j} = 1, \ldots\ldots\ldots, \text{n}_\text{i} \quad (2) \\
Tx_k^{(l)} = \sum_{i=1}^{n} P_{i,j} X_{i,j}^{(l)} & \text{pour k} = 1, \ldots\ldots, \text{m} \;;\; \text{l} = 1, \ldots\ldots, \text{m}_\text{k} \quad (3) \\
t_{k_1}^{l_1} + P_{i,j}\, X_{i,j}^{l_1} \leq t_{k_2}^{l_2} + \text{BS}\left(2 - X_{i,j}^{(l1)} - X_{i,j+1}^{(l2)}\right) & \quad (4) \\
\text{Pour } \text{l1} = 1, \ldots\ldots, \text{mk} \;;\; \text{l2} = 1, \ldots\ldots, \text{m}_\text{k} & \\
t_k^{(l)} = s_k^0 & \text{pour k} = 1, \ldots\ldots, \text{m} \quad (5) \\
t_k^{(r)} = \sum_{l=1}^{r-1} (Tx_k^{(l)} + s_k^l) & \text{pour k} = 1, \ldots\ldots, \text{m} \;;\; \text{r} = 1, \ldots\ldots, \text{m}_\text{k} \quad (6) \\
t_k^{m_k} + Tx_k^{m_k} \leq C_{max} & \text{pour k} = 1, \ldots\ldots, \text{m} \quad (7)
\end{cases}
$$

En plus des données présentées au début de la section dans le cadre général du problème de job shop, le modèle de H.M. Wagner a mis en œuvre l'ensemble des données et variables de décision suivantes :

0 : Opération fictif désignant le début fictif de l'horizon de production

m_k : Nombre d'opérations exécutées sur la machine M_k

s_k^0 : Ecart (durée) entre l'opération sur la machine M_k

$s_k^{(l)}$: Ecart entre la fin de la 1ème opération et le début de la (1+1) ème sur la machine M_k

$Tx_k^{(l)}$: Temps opération de la 1ème opération sur la machine M_k

$t_k^{(l)}$: Date de début de la 1ème opération sur la machine M_k

BS : une borne supérieure de la durée totale de l'ordonnancement. Cette borne peut simplement être la somme de tous les temps opératoires.

$$X_{i,j}^{(l)} = \begin{cases} 0 \; si \; O_{ij} \text{ passe } u_{ij} \text{ en } 1^{ème} \text{ position} \\ 1 \; si \; non \end{cases}$$

Les premières contraintes (1) signifient qu'une unique opération doit passer en $1^{ème}$ position sur la machine M_k. Tandis que les contraintes (2) imposent qu'une opération O_{ij} ne soit affectée qu'à une position et une seule.

Les contraintes (3) déterminent le temps opératoire de l'éme opération exécutée sur la machine M_k. Les contraintes (4) expriment la relation de précédence imposée entre deux opérations O_{ij} et $O_{i\,j+1}$.

Le calcul des dates de début au plus tôt des opérations sur les machines, est donné par les contraintes (5) et (6). Alors que les dernières contraintes (7) permettent de calculer le makespan C_{max} .

Exemple d'application :

On considère le problème suivant de job shop à trois jobs et trois machines :

Jobs	La séquence de machine	**Les durées**
1	**1, 2, 3**	**p_{11}=4, p_{21}=3, p_{31}=3**
2	**1, 3,2**	**P_{12}=1, p_{32}=5, p_{22}=3**
3	**2, 1, 3**	**P_{23}=1, p_{13}=5, p_{33}=3**

- **Modélisation :**

- **Variables:**
  - **t_{ij} = la date de début de job j sur la machine i.**
  - **x_{ijk} = 1 si le job j précède le job k sur la machine i.**
    0 si non.
  - **M une borne supérieure de la duré totale de l'ordonnancement.**

- **le programme mathématique :**

Min C_{max}

Sujet à

- **$C_{max} \geq t_{31} + p_{31}$**

- $C_{max} \geq t_{22} + p_{22}$
- $C_{max} \geq t_{33} + p_{33}$
- $t_{21} \geq t_{11} + p_{11}$
- $t_{31} \geq t_{21} + p_{21}$
- $t_{32} \geq t_{12} + p_{12}$
- $t_{32} \geq t_{32} + p_{32}$
- $t_{13} \geq t_{23} + p_{23}$
- $t_{33} \geq t_{13} + p_{13}$

Pour i=1, 2, 3.

- $t_{i1} + p_{i1} \leq t_{i2} + M\,(1-x_{i12})$
- $t_{i2} + p_{i2} \leq t_{i1} + M\,x_{i12}$
- $t_{i1} + p_{i1} \leq t_{i3} + M\,(1-x_{i13})$
- $t_{i3} + p_{i3} \leq t_{i1} + M\,x_{i13}$
- $t_{i2} + p_{i2} \leq t_{i3} + M\,(1-x_{i23})$
- $t_{i3} + p_{i3} \leq t_{i2} + M\,x_{i23}$

$t_{ij} \geq 0$, **pour i=1, 2, 3 et j=1, 2, 3**

$x_{ijk} \in \{0,1\}$ **pour i = 1, 2, 3 j =1, 2, 3 et j < k**

- ✓ Les contraintes noires permettent de calculer le Makespan C_{max} .
- ✓ Les contraintes bleues permettent de garder la gamme opératoire pour chaque job.
- ✓ Les contraintes rouges signifient que chaque machine ne peut exécuter qu'une seule opération à la fois.

Ce type d'atelier peut être représenté aussi par un graphe potentiel tâches noté $G(X;U)$, où X est l'ensemble des sommets et U l'ensemble des arcs.

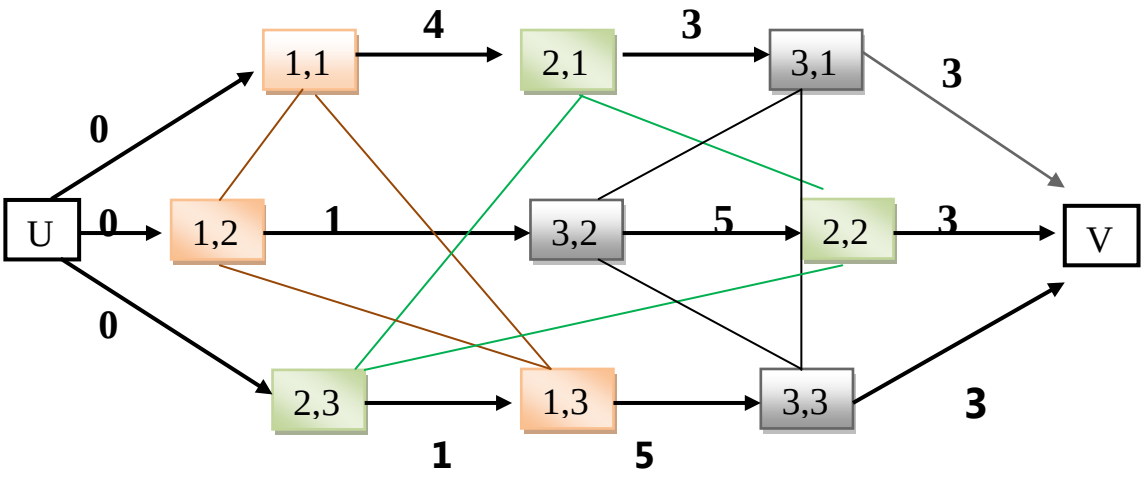
Si on appelle T l'ensemble des n tâches du problème, X est alors la réunion de T et de l'ensemble (0,n+1) où (0) représente la tâche du début fictif et {n+1} la tâche de la fin fictive [5].

L'ensemble des arcs U est {(0 ; i), (i ; j) et (i ; n+1) et j $\in T$} .Il représente les différentes contraintes reliant les tâches : les contraintes de précédence sont représentées par des arcs conjonctifs et les contraintes de ressources par des arcs disjonctifs [5].

Les valeurs portées par les arcs dépendent des sommets :

- l'arc de type (0; i) : porte la date de disponibilité de la tâche i.

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



Arc disjonctif (entre les tâches de machines M_1)

Arc disjonctif (entre les tâches de machines M_2)

Arc disjonctif (entre les tâches de machines M_3)

Arc conjonctif.

U : la tâche du début fictif.

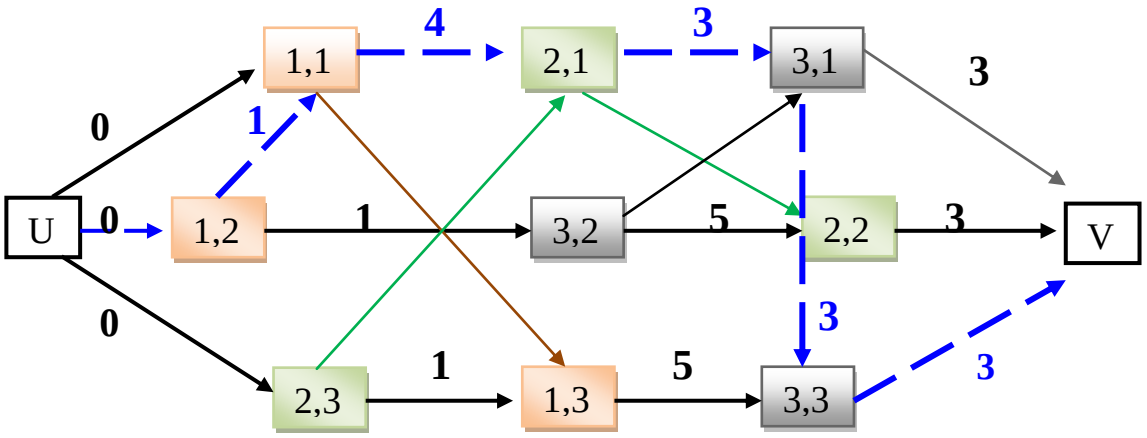
V : la tâche de la fin fictive.

1.2. Optimisation :

1.2.1. Job shop à m >2 machines:

Pour obtenir un ordonnancement réalisable de problème précédant il faut orienter les arcs disjonctifs de façon à former un graphe acyclique. Son « makespan » représente la longueur du plus long chemin allant du nœud U au nœud V.

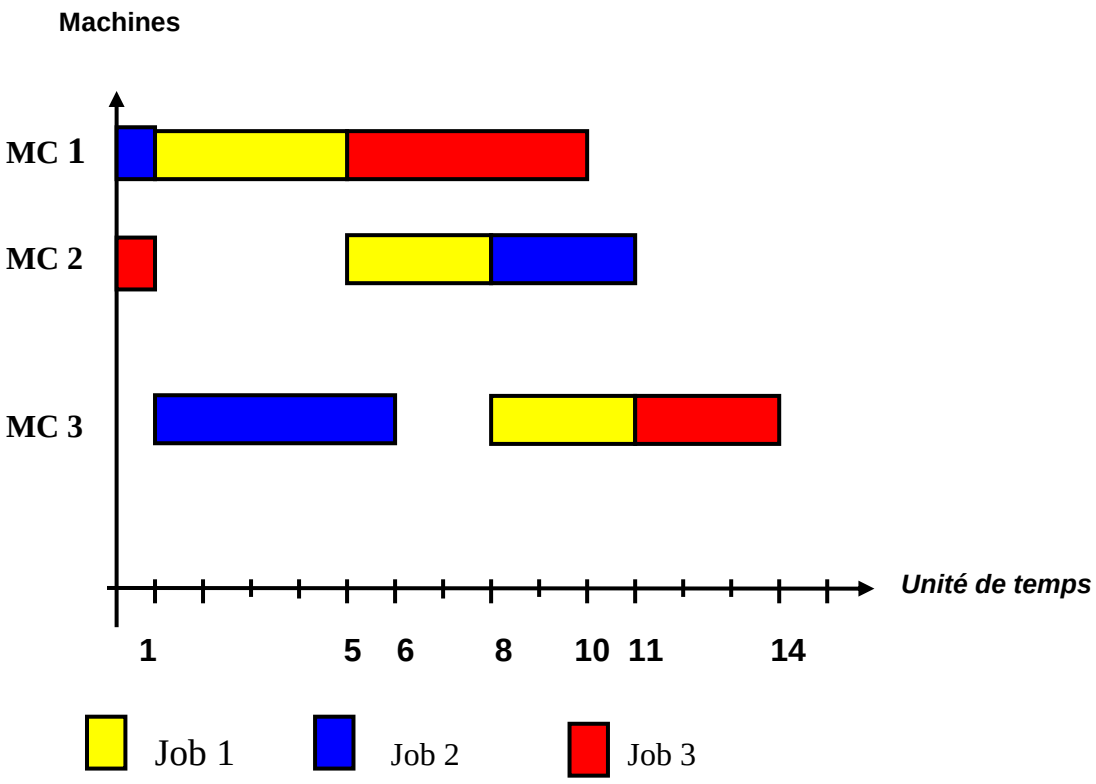
-Exemple d'un ordonnancement réalisable :



La longueur de chemin bleu représente la valeur de makespan associé à cet ordonnancement, alors le $\mathbf{C_{max}}$ =14.

-Représentation de l'ordonnancement par le diagramme de Gantt :

Le diagramme de Gantt présente en ordonnées la liste des tâches, notées T_i à exécuter par les machines, notées M_j, et en abscisses l'échelle du temps, comme le montre la figure suivante dans le cas de la solution précédente [2].



1.2.2. Job shop à 2 machines:

On peut résoudre $n/2/J/C_{max}$ à partir un algorithme, dû à «***JACKSON***», constitue une extension de celui de JOHNSON. Il traite de plus le cas où certains jobs peuvent n'avoir à passer que sur une des deux machines. Notant M_1 et M_2 les deux machines, on peut alors partitionner l'ensemble des jobs en 4 classes : C_1 : ensemble des jobs ne passant que sur M1, C_2 : ensemble des jobs ne passant que sur M_2, C_{12} : ensemble des jobs passant sur M_1 puis sur M_2, C_{21} : ensemble des jobs passant sur M_2 puis sur M_1. On montre alors qu'un ordonnancement optimal peut être déterminé en adoptant le séquencements :

- **sur M1 : C_{12}-C_1-C_{21}**
- **sur M2 : C_{21}-C_2-C_{12}**

Où les jobs de C12 et C21 sont ordonnancés par la règle de JOHNSON.

Algorithme de JOHNSON :

- Partitionner l'ensemble des tâches en 2 sous-ensembles :
  - $A = \{j \in J, a_j \leq b_j\}$
  - $B = \{j \in J, a_j > b_j\}$
- Séquencer les jobs de A dans l'ordre croissant des a_j
  ➔ Séquence π_A
- Séquencer les jobs de B dans l'ordre décroissant des b_j
  ➔ Séquence π_B
- Fusionner les deux séquences ➔ $\pi_o = \pi_A, \pi_B$
- On utilise le même ordre de passage sur les deux machines

Exemple :

J_i	J_1		J_2		J_3		J_4		J_5		J_6	
$O_{i,j}$	O_{11}	O_{21}	O_{12}	O_{22}	O_{13}	O_{23}	O_{14}	O_{24}	O_{15}	O_{25}	O_{16}	O_{26}
M_k	M_1	M_2	M_1	-	M_1	M_2	M_2	M_1	-	M_2	M_1	M_2
P_{ij}	3	4	7	-	6	3	4	5	-	4	5	4

J_i	J_7		J_8		J_9	
$O_{i,j}$	O_{17}	O_{17}	O_{18}	O_{18}	O_{19}	O_{19}
M_k	M_2	M_1	M_1	M_2	M_2	M_1
P_{ij}	4	2	2	4	6	5

D'après la méthode de «***JACKSON*** », l'ordonnancement optimale sur :

- **M_1 est C_{12}-C_1-C_{21}**
- **M_2 est C_{21}-C_2-C_{12}**

Où :

- **C_1** l'ensemble des jobs qui utilisent seulement la machine 1 $\{J_2\}$
- **C_{12}** l'ensemble des jobs qui utilisent la machine 1 puis la machine 1 $\{J_1,J_3,J_6, J_8\}$ on va les ordonnancer avec la méthode de ***JOHNSON***
- **C_{21}** l'ensemble des jobs qui utilisent la machine 2 puis la machine 1 $\{J_4, J_7, J_9\}$ on va les ordonnancer avec la méthode de ***JOHNSON***
- **C_2** l'ensemble des jobs qui utilisent seulement la machine 2 $\{J_5\}$

✓ L'**ordonnancement de C_{12} :**

- $A = \{j \in J, a_j \leq b_j\}$ ➔ $A=\{1, 8\}$
- $B= \{j \in J, a_j > b_j\}$ ➔ $B = \{3,6\}$
- π_A la séquence des taches de A dans l'ordre croissant des a_j

  ➔ $\pi_A= J_1 - J_8$
- π_B la séquence des taches de B dans l'ordre décroissant des b_j

  ➔ $\pi_B= J_6 - J_3$

➢ La séquence optimale $\pi_{C\ 12}= \{ J_1, J_8, J_6, J_3\}$

✓ **L'ordonnancement de C_{21} :**

- $A = \{j \in J, a_j \leq b_j\}$ ➔ $A=\{4\}$
- $B= \{j \in J, a_j > b_j\}$ ➔ $B = \{7,9\}$

- π_A la séquence des taches de A dans l'ordre croissant des a_j

  ➔ $\pi_A = J_4$

- π_B la séquence des taches de B dans l'ordre décroissant des b_j

  ➔ $\pi_B = J_9 - J_7$

➢ La séquence optimale $\pi_{C\,21} = \{ J_4, J_9, J_7 \}$

➢ ***L'ordonnancement optimal de problème est :***

- **Sur M_1 : $\pi_{C\,12}$- J_1- $\pi_{C\,21}$**
- **sur M_2 : $\pi_{C\,21}$- J_5- $\pi_{C\,12}$**

2. Flow shop classique :

Nous rappelons que dans un atelier de Flow shop toutes les opérations passent par toutes les machines dans le même et unique ordre.

2.1. Exemple d'une modélisation de Flow shop :

On considère le problème suivant de Flow shop à 3 machines et 4 jobs :

	M_1	M_2	M_3
	P_{i1}	P_{i2}	P_{i3}
J_1	2	5	1
J_2	4	3	2
J_3	6	1	4
J_4	4	2	2

- **Modélisation :**

- **Variables:**
  - t_{ij} = **la date de début de job i sur la machine j.**

- **le programme mathématique :**

Min C_{max}

Sujet à

- $C_{max} \geq t_{13} + p_{13}$
- $C_{max} \geq t_{23} + p_{23}$
- $C_{max} \geq t_{33} + p_{33}$
- $C_{max} \geq t_{43} + p_{43}$
- $t_{12} \geq t_{11} + p_{11}$
- $t_{13} \geq t_{12} + p_{21}$
- $t_{22} \geq t_{21} + p_{21}$
- $t_{23} \geq t_{22} + p_{22}$
- $t_{32} \geq t_{31} + p_{31}$
- $t_{33} \geq t_{32} + p_{32}$
- $t_{42} \geq t_{41} + p_{41}$
- $t_{43} \geq t_{42} + p_{42}$
- $t_{i1} + p_{i1} \leq t_{j1}$ **ou** $t_{j1} + p_{j1} \leq t_{i1}$
- $t_{i2} + p_{i2} \leq t_{j2}$ **ou** $t_{j2} + p_{j2} \leq t_{i2}$ **i=1,2,3 et j=1,2,3 avec** $i \neq j$
- $t_{i3} + p_{i3} \leq t_{j3}$ **ou** $t_{j3} + p_{j3} \leq t_{i3}$
- $t_{ij} \geq 0$ **i=1,2,3,4 j=1,2,3.**

✓ Les contraintes noires permettent de calculer le Makespan C_{max} .

✓ Les contraintes bleues sont les contraintes de précédences

✓ Les contraintes au bleu ciel son les contraintes de ressources qui signifient que chaque machine ne peut exécuter qu'une seule opération à la fois.

2.2. Méthodes d'optimisation :

2.2.1 Flow shop à m>2 machines:

NEH (Nawaz-Enscore-Ham) est une heuristique très connue pour la résolution des problèmes d'ordonnancements de type Flow shop.

Le principe est le suivant :

- Trier les jobs par $\sum_{j=1}^{m}(P_{ij})$ croissant.
- Ordonnancer les deux premiers travaux dans l'ordre minimisant le C_{max}.
- Ensuite, prendre chaque autre travail dans la liste triée et l'insérer à la meilleure position pour la minimisation du C_{max} dans l'ordonnancement partiel.

Afin d'illustrer cette heuristique, on considère le problème précédant de Flow shop à 3 machines et 4 jobs :

	M_1	M_2	M_3	
	P_{i1}	P_{i2}	P_{i3}	$\sum_{i=1}^{m} Pij$
J_1	2	5	1	8
J_2	4	3	2	9
J_3	6	1	4	11
J_4	4	2	2	8

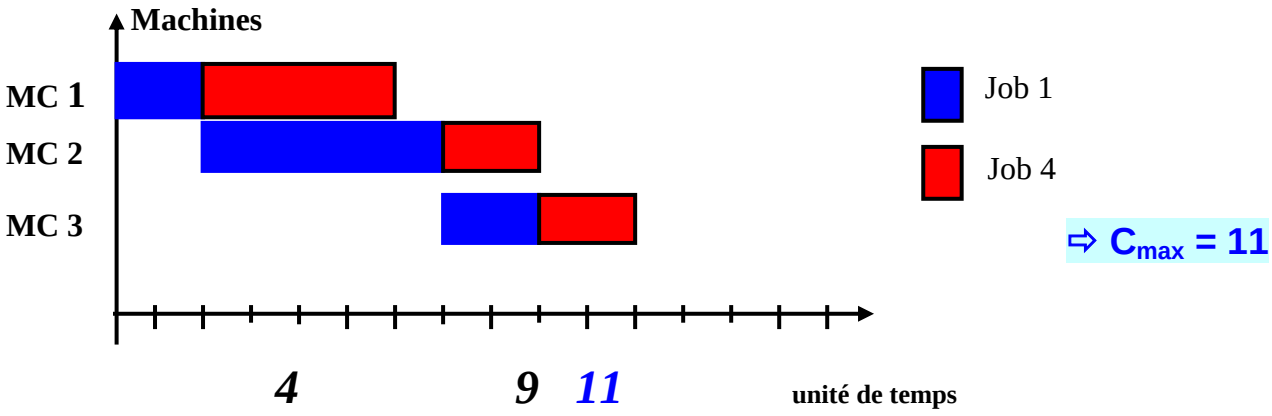
- ***On utilise la méthode de NEH :***

- On trie les jobs par $\sum_{j=1}^{m}(P_{ij})$ croissant.

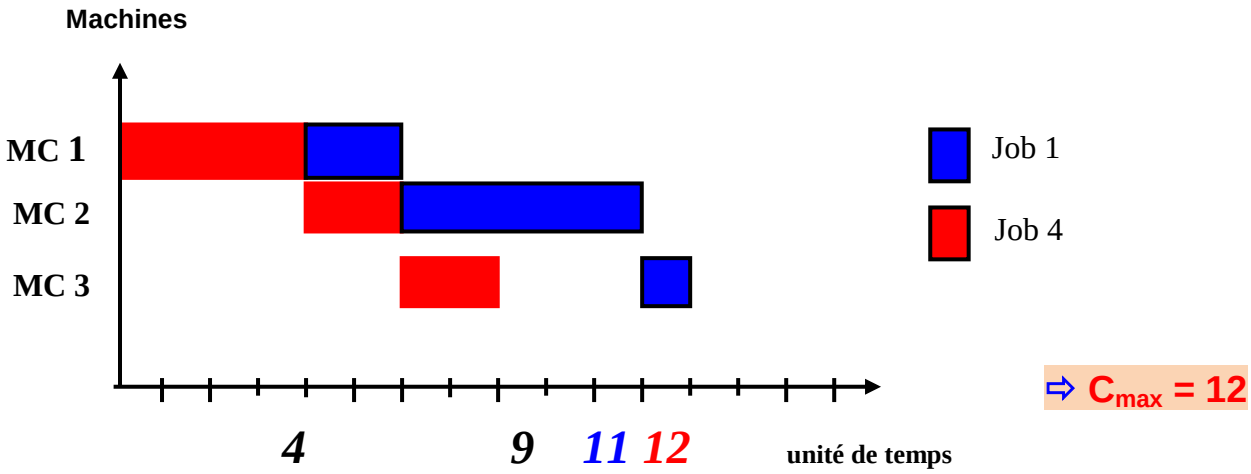
J_1	J_4	J_2	J_3

- On ordonnance les deux premiers travaux dans l'ordre minimisant le C_{max}.

❖ **(J_1, J_4) :**



❖ (J_4, J_1) :



⇨ $C_{max} = 12$

➔ **Donc le job 1 doit exécuter avant le job 4.**

– Ensuite, on prend chaque autre travail dans la liste triée et l'insérer à la meilleure position pour la minimisation du C_{max} dans l'ordonnancement partiel.

- *Insérer le J_2 :*

(J_2,J_1,J_4) ⇨ $C_{max} = 16$
(J_1,J_2,J_4) ⇨ $C_{max} = 14$
(J_1,J_4,J_2) ⇨ $C_{max} = 15$

- *Insérer le J_3 :*

(J_3,J_1,J_2,J_4) ⇨ $C_{max} = 20$
(J_1,J_3,J_2,J_4) ⇨ $C_{max} = 20$
(J_1,J_2,J_3,J_4) ⇨ $C_{max} = 20$
(J_1,J_2,J_4,J_3) ⇨ $C_{max} = 21$

⟶ **Il y a 3 séquences possibles à retourner.**

2.2.2 Flow shop à 2 machines:

JOHNSON proposa, en 1954, un algorithme optimal pour le $n/2/F/C_{max}$ .On applique, pas à pas la règle suivant :

- Partitionner l'ensemble des tâches en 2 sous-ensembles :
  - $A = \{j \in J, a_j \leq b_j\}$
  - $B = \{j \in J, a_j > b_j\}$
- Séquencer les jobs de A dans l'ordre croissant des a_j

  ➔ Séquence π_A
- Séquencer les jobs de B dans l'ordre décroissant des b_j

  ➔ Séquence π_B
- Fusionner les deux séquences ➔ $\pi_o=\{\pi_A, \pi_B\}$
- On utilise le même ordre de passage sur les deux machines

Exemple :

Jobs	*La duré sur la machine 1 (a_j)*	*La duré sur la machine 2 (b_j)*
1	*6*	*8*
2	*4*	*5*
3	*4*	*1*
4	*8*	*4*

- $A= \{j \in J, a_j \leq b_j\}$

  ➔ $A = \{1,2\}$
- $B= \{j \in J, a_j > b_j\}$

  ➔ $B = \{3,4\}$
- $\pi_1= 2 - 1.$
- $\pi_2 = 4 - 3.$

La séquence optimale est : $\pi_O=\{2,1,4,3\}$

On va dessiner le digramme de Gantt de l'ordonnancement pour obtenir la valeur optimale de C_{max}.

⇨ La valeur optimale de C_{max} est 23.

V. Conclusion

Dans ce projet, nous avons présenté les différentes caractéristiques d'un problème d'ordonnancement et les différents types de problèmes d'ordonnancement d'atelier.

Nous avons traité en particulier le problème de flow shop et de job shop, mais la difficulté réside dans la résolution d'un problème d'ordonnancement par les méthodes heuristiques et méta-heuristiques ce qui nécessite une étude approfondie sur ces méthodes approchés

Bibliographie :

1. ***AHMED EL HILALI ALAOUI et AL*** *«* Initialisation à la recherche opérationnelle »,2009.

2. ***ASMA KARRAY*** « Contribution à l'ordonnancement d'ateliers agroalimentaires utilisant des méthodes d'optimisation hybrides », le 05 Juillet 2011, Thèse en vue de l'obtention du garde de docteur en Spécialité : Automatique et Informatique Industrielle.

3. ***FATIMA EL KHOUKHI*** **«** Méta-heuristiques hybrides pour la résolution de problèmes d'ordonnancement de type job shop **»,** *le 21 décembre 2009,* Thèse en vue de l'obtention du garde de docteur de l'Université SIDI MOHAMED BEN ABDLLAH et de l'Université du HAVARE.

4. ***HABIBA HOUARI, «*** Planification et ordonnancement en temps réel d'un job shop en utilisant l'Intelligence Artificielle ***»,*** *le 02 juillet 2012,* Mémoire de Magister en Automatique, Modélisation et commande des systèmes.

5. ***IMEN MEDHBI BRINIS, «*** Ordonnancement d'ateliers de traitements de surfaces pour une production mono-robot/multi-produits : Résolution et étude de la robustesse ***»,*** *le 16 mai 2014,* Thèse en vue de l'obtention du garde Docteur en Spécialité « Automatique, génie informatique, traitement du signal et images »

6. ***MEBAREK KEBABLA, «*** Utilisation des stratégies méta-heuristiques pour l'ordonnancement des ateliers de type job shop ***»,*** le 02 juillet 2008, Mémoire en vue de l'obtention du Diplôme de Magister, présenté au laboratoire d'Automatique et Productique LAP.

7. ***Mohamed Ali ALOULOU, «*** Introduction aux problèmes d'ordonnancement ***»,*** *le 28 novembre 2005,* LAMSADE, Université Paris Dauphine

8. ***SAMIA OURARI,*** «L'ordonnancement déterministe à l'ordonnancement distribué sous incertitude ***»,*** *le 28 janvier 2011, Thèse* en vue de l'obtention du garde de docteur de l'Université de TOULOUSE.

9. ***WAJDI TRABELSI***, « Ordonnancement des systèmes de production flexibles soumis à différents types de contraintes de blocage *», le 14 novembre 2012.*