REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE MOHAMED BOUDIAF - M'SILA

FACULTE mathématique et informatique
DEPARTEMENT informatique
N° :……..

DOMAINE :mathématique et informatique
FILIERE : informatique
OPTION : IDO

Mémoire présenté pour l'obtention
Du diplôme de Master Académique

Par: LANNAK HANANE

Intitulé

Adaptation de la méthode coucou pour la résolution du problème job shop

Soutenu devant le jury composé de :

HAMMAK ALAWA	Université de M'sila	Président
LAMICHE CHAABANE	Université de M'sila	Rapporteur
ABACHA FARID	Université de M'sila	Examinateur

Année universitaire : 2018 /2019

Remerciements

Tout d'abord El-hamdoulillah, c'est lui qui m'a donné l'effort, le temps et (el-taofik) pour réaliser ce mémoire.

J'adresse mes plus vifs remerciements à Monsieur Dr. *Ch.lamiche* pour avoir accepté de me suivre dans l'élaboration de ce travail et pour leurs conseils et leur disponibilité pendant toute la durée de la réalisation de ce mémoire.

Un remerciement très chaleureux à mes parents et toute ma famille pour leur présence constante à mes côtés et leur soutien.

Je tiens à remercier tous les enseignants et les étudiants de informatique décisionnelle et optimisation qui ont ma donnée des aides importantes.

« On a toujours assez du temps si on l'exploite bien»

TABLE DES MATIERES

CHAPITRE 2 : LE PROBLEME JOB SHOP

CHAPITRE 4 : RESULTATS EXPRIMENTAUX

Liste des Tableaux

Liste des figures

INTRODUCTION GENERALE

L'optimisation joue un rôle très essentiel dans la recherche opérationnelle, informatique et mathématique. Elle consiste à ajuster des entrées selon des caractéristiques souhaitées. En fait, l'optimisation est un processus mathématique ou expérimental permettant d'aboutir à un résultat minimal ou maximal. Dans un problème d'optimisation, l'entrée est un ensemble de variables et de contraintes, le traitement est un ensemble d'opérations permettant la satisfaction d'une fonction objectif (connue aussi par: fonction fitness) et la sortie est une solution qui a une certaine qualité calculée selon la fonction objectif du problème traité. De nombreux algorithmes de résolution de problèmes d'optimisation ont été proposés dans la littérature

les problèmes d'ordonnancement des ateliers ont été très étudiés dans la littérature depuis plus de 50 ans. Ils sont définis par un ensemble de travaux à réaliser sur un ensemble de ressources ; de sorte qu'une fonction objectif soit optimisée. Ils sont souvent classés NP-Difficiles. Leur résolution nécessite des méthodes dédiées à leur degré de complexité ; pour cette raison plusieurs heuristiques et méta-heuristiques ont été conçues.

Dans cette mémoire : nous avons développé une approche basé sur la recherche Coucou pour la résolution du problème job shop afin de minimisons la date de fin de toutes les opérations. Dans cette approche nous proposons une nouvelle représentation du nid et différentes stratégie de tri.

Cette mémoire est organisée en quatre chapitres comme suit :

Dans **Le premier chapitre** nous donnons une généralité sur les problèmesd'ordonnancement (ses notions de base, leurs différentes classes et leurs principales caractéristiques et l'ensemble des méthodes de résolution utilisées).

Dans **le deuxième chapitre** nous avons abordé la description du problème d'ordonnancement job shop.

Dans **Le troisième chapitre** nous présentons une méta heuristique « les algorithmes de recherche coucou (Cuckoo Search algorithme) pour résoudre le problème d'ordonnancement d'un système de production de type Job-Shop afin de minimiser le makespan.

Ce chapitre contient deux parties. Dans la première partie, nous présentons les algorithmes de recherche coucou et ses principes. Dans la deuxième partie nous appliquons cette nouvelle méta heuristique pour la résolution des problèmes d'ordonnancement job shop classique

Dans **le quatrième chapitre**, nous présentons de donner une synthèse des résultats obtenus par application la méta heuristique(recherche coucou) a la résolution d'un ensemble de problèmes tests de job shop .afin d'évaluer les performances des différents choix implémentes algorithme coucou Nous terminons la présentation de cette mémoire par une conclusion

CHAPITRE 1

ELEMENT FONDAMONTEUX DE L'ORDONNANCEMENT

1 Introduction

L'ordonnancement est une branche de la recherche opérationnelle et de la gestion de la production qui vise à améliorer l'efficacité d'une entreprise en termes de coûts de production et de délais de livraison. Les problèmes d'ordonnancement apparaissent dans tous les domaines de l'économie :l'informatique(tâches :jobs ;ressources :processeurs ou mémoire...),la construction(suivi de projet),l'industrie(problèmes d'ateliers, gestion de production),l'administration(emplois du temps).

Dans ce chapitre, quelques généralités, sur les problèmes l'ordonnancement en commençant par définir ce qu'est un problème, ses caractéristiques, notation et solution. Nous intéressons à la complexité des problèmes d'ordonnancement et plus généralement, à la théorie de la complexité. , les méthodes exactes de résolution, à l'instar de la programmation linéaire et du Branch and Bound, sont définies. Nous avons aussi introduit les méthodes approchées.

2 Définition du problème d'Ordonnancement

De nombreuses définitions du problème d'ordonnancement sont proposées dans la Littérature. Parmi les définitions les plus souvent rencontrées sont :« Un problème d'ordonnancement est défini par un ensemble de jobs à réaliser sur un ensemble de ressources ; de sorte qu'une fonction objectif soit optimisée » [1]. Ordonnancer c'est programmer l'exécution d'une réalisation en attribuant des ressources aux tâches et en fixant leurs dates d'exécution ». [2].

Tout problème d'ordonnancement se définit donc par un ensemble de tâches, encore appelées activités, à exécuter sur un ensemble de ressources. Il s'agit d'affecter, à chaque activité, une date d'exécution et un sous-ensemble de ressources de façon à satisfaire à la fois des contraintes temporelles (par exemple, respecter un ordre partiel sur l'ensemble des

activités) et des contraintes d'accès aux ressources (par exemple, exécuter sur une ressource au plus une activité à chaque instant). Enfin, la programmation des activités doit généralement répondre à un critère d'optimisation (par exemple, minimiser la durée totale de l'ordonnancement). Les problèmes d'ordonnancement touchent tous les domaines de l'économie : [1].

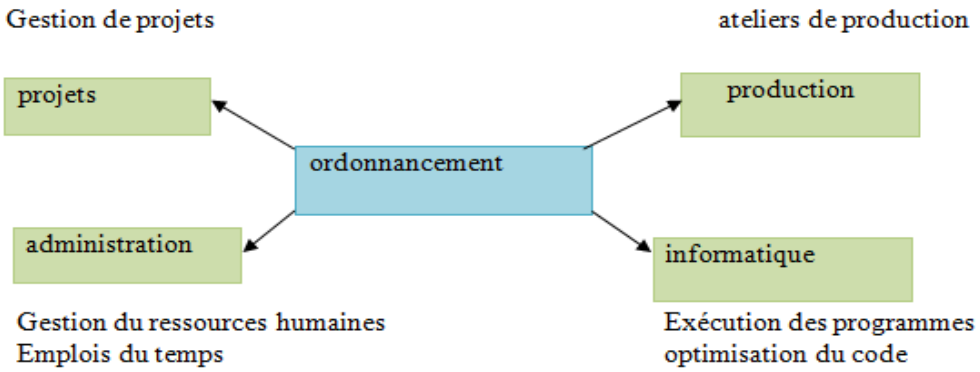

Figure1.1 **:** les domaines des problèmes d'ordonnancement.

3 Ordonnancement et typologie organisationnelle

En gestion de production, pendant très longtemps, l'objectif principal a été la recherche d'une plus grande efficacité de l'outil de production, elle était, à l'image de l'organisation de l'entreprise toute entière, structurée par fonctions ou services, spécialisés et liés hiérarchiquement. En fait, cette hiérarchie obéit à un modèle de résolution de problème, décomposant les décisions en trois niveaux [3][4][5]:

- le niveau stratégique :

Il s'agit de formulation de la politique et la stratégie de l'entreprise à long terme, qui porte essentiellement sur la gestion des ressources durables, afin que celles-ci soient en mesure d'assurer la pérennité de l'entreprise sur plusieurs années .

- le niveau tactique :

Il s'agit de décisions à moyen terme qui s'effectuent sur plusieurs moins. Elles assurent la liaison entre le niveau stratégique et le niveau opérationnel .l'objectif est de produire au moindre cout pour satisfaire la demande prévisible, en s'inscrivant dans le cadre fixé par le plan stratégique de l'entreprise.

- **le niveau opérationnel :**

C'est le niveau d'exécution du système qui sert à une gestion quotidienne pour faire face à la demande du jour le jour ,dans le respect des décisions tactiques .il génère des décisions à court et à très court terme qui déterminent l'ordonnancement et le pilotage.

4 Les Différents éléments d'un problème d'Ordonnancement

D'âpres la définition de l'ordonnancement, nous en déduisons qu'un ordonnancement est constitué principalement de quatre élément suivants :

Les tâches, les ressources, les contraintes et les objectifs ou les critères d'optimisation.

4.1 Les tâches

Une tâche Tâche(i) est, par définition, une entité élémentaire localisée dans le temps par une date de début (Si) et/ou de fin (Ci), et par une durée d'exécution (pi) ; par ailleurs, les tâches nécessitent l'utilisation de ressources (voir Figure.2).Les tâches sont, le plus souvent, liées entre elles par des conditions d'origines diverses ; on dit alors que les tâches sont interdépendantes. Dans le cas contraire, elles sont dites indépendantes.

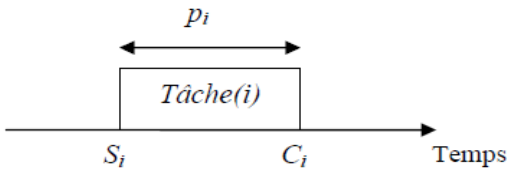

Figure 1.2 :une tâche.

Dans certains problèmes, les tâches peuvent être interruptibles ce qui signifie, en pratique, qu'elles peuvent être exécutées par morceaux par une ou plusieurs ressources ; dans

d'autres cas contraires, on ne peut interrompre une tâche une fois celle-ci commencée. On parle respectivement de problèmes préemptif et non préemptif. [6][7].

4.2 Les ressources

Une ressource est un moyen technique ou humain destinée à être utilisée pour la réalisation d'au moins une tâche et disponible en quantité limitée.

Les ressources peuvent être classées soit selon leurs disponibilités au cours du temps, et on parle dans ce cas de ressources renouvelables ou consommables, soit selon leurs capacités et ceux sont des ressources disjonctives (non partageables) ou cumulatives (partageables).

- Ressource consommable: il s'agit d'une ressource dont la disponibilité décroît après avoir été allouée à une tâche, par exemple la matière première.

- Ressource renouvelable: il s'agit d'une ressource pouvant redevenir disponible en même quantité après avoir été allouée à une tâche (les machines, les hommes, l'équipement,…).

Pour les ressources renouvelables :

- Ressource disjonctive (non partageables): il s'agit d'une ressource qui ne peut exécuter qu'une seule tâche à la fois.

- Ressource cumulative (partageables): il s'agit d'une ressource qui peut être utilisée simultanément par plusieurs tâches.

4.3 Les contraintes

Une contrainte est une restriction sur les valeurs que peuvent prendre une ou plusieurs variables de décision sur le temps (variable d'ordonnancement) ou bien sur les ressources (variables d'affectation) On distingue :

- Les contraintes potentielles

Elles peuvent être de deux sortes :

Les contraintes d'antériorité selon laquelle une tâche j ne peut commencer avant une tâche i ne soit terminée, par exemple, la construction des piliers suit les fondations.

Les contraintes de localisation temporelle impliquant qu'une tâche donnée i ne peut débuter avant une date imposée, ou qu'elle peut s'achever après une date imposée.

- Les contraintes disjonctives

Une contrainte disjonctive impose la non-réalisation simultanée de deux tâches A et B. On trouve de telles contraintes dans le cas d'utilisation d'une ressource présente en un seul exemplaire (une grue, une équipe, etc.) ou pour formuler des interdictions de réalisation simultanée pour des raisons de sécurité ou des problèmes de place. Arbitrer une contrainte disjonctive consiste à décider si A sera fait avant B ou l'inverse.

- Les contraintes cumulatives lorsque les tâches demandent une partie d'une ou plusieurs ressources présentes en quantité limitée.

Le problème est beaucoup plus combinatoire que pour les contraintes disjonctives

4.4 Les objectifs

Résoudre un problème d'ordonnancement c'est avant tout choisir ce que l'on veut optimiser. Selon le critère que l'on cherche a minimiser ou maximiser, on peut choisir entre deux grands types de stratégies, visant respectivement a l'optimalité des solutions, ou plus simplement à leur admissibilité. Comme critères, on notera particulièrement ceux :

- lies au temps :
  - Le temps total d'exécution ou le temps moyen d'achèvement d'un ensemble de taches .
  - Le stock d'en-cours de traitement .
  - Différents retards (maximum, moyen, somme, nombre, etc.)ou avances par rapport aux dates limites fixées .
- lies aux ressources :
  - La quantité totale ou pondérée de ressources nécessaires pour réaliser un ensemble de taches .

- la charge de chaque ressource .

➢ lies a une énergie ou un débit .

➢ lies aux couts de lancement, de production, de transport, etc..

4.5 Les critères

Un critère correspond à des exigences qualitatives et quantitatives à satisfaire permettant d'évaluer la qualité de l'ordonnancement établi.

- Les critères dépendant d'une application donnée sont très nombreux ; plusieurs critères peuvent être retenus pour une même application. Le choix de la solution la plus satisfaisante dépend du ou des critères préalablement définis, pouvant être classés suivant deux types ,réguliers et irréguliers.

Les différents critères ne sont pas indépendants; certains même sont équivalents. Deux critères sont équivalents si une solution optimale pour l'un est aussi optimale pour l'autre et inversement [2]

➢ Les critères réguliers constituent des fonctions décroissantes des dates d'achèvement des opérations. Quelques exemples sont cités ci-dessous:

- 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.

➢ Les critères irréguliers sont des critères non réguliers, c'est-à-dire qui ne sont pas des

fonctions monotones des dates de fin d'exécution des opérations, tels que:

- 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 [8] et à la recherche de solutions à des problèmes complexes d'optimisation.

5 Classification des problèmes d'ordonnancement

Il existe une très grande variété de problèmes d'ordonnancement. Pour leur identification et leur classification , nous adoptons la notation proposée par Graham et al [1979]. Cette notation est constituée de trois champs $\alpha|\beta|\gamma$. [9]

- Le premier champ α est constitué de deux éléments : $\alpha = \alpha 1 \alpha 2$ il décrit l'environnement des machines utilisées :

- le paramètre $\alpha 1 \epsilon$ {∅, P, Q, R, F, J, O} décrit le type des machines utilisées tel que :

- Une machine (∅) **:**Un problème est dit problème à une machine (ou ressource unique) si on ne dispose qu'une seule ressource pour réaliser un ensemble de travaux. Ces derniers sont constitués d'une seule tâche pour chacun d'eux. Dans ce type de problème la résolution consiste à trouver l'ordre optimal d'exécution de ces tâches vis-à-vis d'un critère donné.

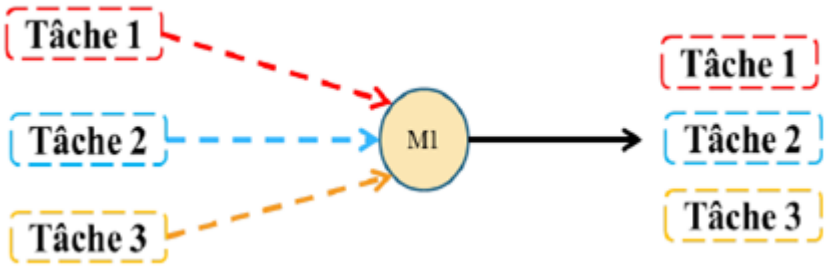

Figure1.3:machine unique.

- machines parallèles : Les problèmes d'ordonnancement à machines parallèles se caractérisent par l'existence de plus d'une ressource. Elles représentent un cas particulier

de problèmes multi-machines où les machines sont disposées en parallèle. On peut distinguer trois types des problèmes à machines parallèles:

- machines identiques (P) : les durées opératoires sont égales et ne dépendent donc pas des machines.
- machines uniformes (Q) : la durée d'une opération varie uniformément en fonction de la performance de la machine choisie,
- machines indépendantes (R) : les durées opératoires dépendent complètement des machines utilisées.

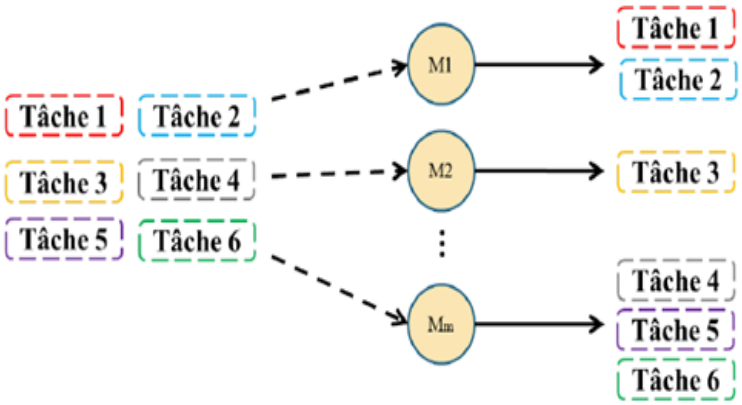

Figure1.4 :exemple de machines parallèles.

- Les ateliers de type flow-shop (F) :Appelés également ateliers à cheminement unique, ce sont des ateliers où une ligne de fabrication est constituée de plusieurs machines en série ; toutes les opérations de toutes les tâches passent par les machines dans le même ordre.

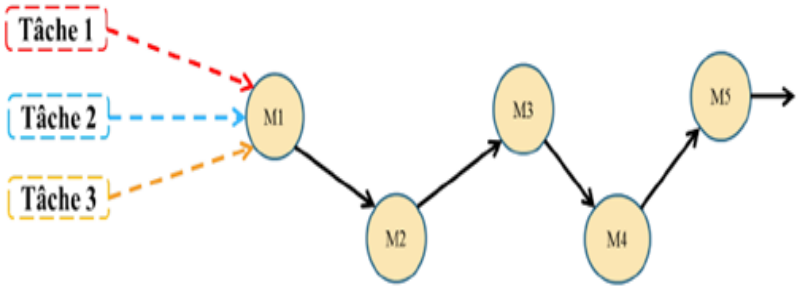

Figure 1.5 :exemple de type flow shop.

- Les ateliers de type job-shop (J) : Le problème de Job Shop consiste à réaliser un ensemble de n Job (travail) sur un ensemble de m machines en cherchant d'atteindre certain objectifs. Chaque Job j est composé de n_j tâches devant être exécutées sur les différentes machines selon un ordre préalablement défini. Les travaux ne s'exécutent pas sur toutes les machines et de plus n'ont pas le même ordre d'exécution. En effet chaque travail emprunte le chemin qui lui est propre.

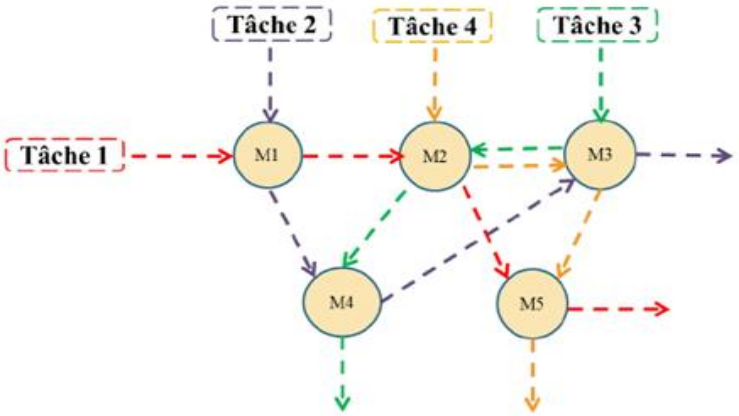

Figure 1.6 :exemple de type job shop.

- Les ateliers de type open-shop (O) : Dans le problème de type open shop l'ordre d'exécution de tâches d'un travail est totalement libre, c'est-à-dire aucune contrainte est remarquée sur la précédence entre tâches. Les produits passent alors dans n'importe quelle direction (les gammes sont libres).

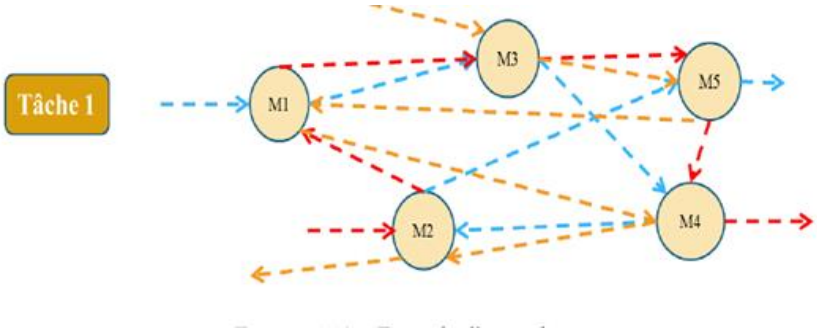

Figure 1.7:exemple de type open shop.

Le deuxième champ $\beta = \beta1\beta2\beta3\beta4\beta5\beta6\beta7$décrit les caractéristiques des travaux et des machines. Il est utilisé pour préciser les contraintes du problème et les différentes hypothèses sur le mode d'exécution des tâches (préemption, précédence, etc.).

- Le dernier champ γ indique le critère d'optimisation, il peut donc prendre de nombreuses valeurs et peut être une combinaison de plusieurs critères.

A titre d'exemple la notation : J,5,9|$Prec$,ri| $Cmax$

Signifie qu'il s'agit d'un problème d'ordonnancement d'un atelier de type job-shop (J), de cinq jobs à neuf machines(5,9). Les jobs présentent une contrainte de précédence, (Prec), et une contrainte ri de dates de début au plus tôt. Avec pour objectif est de minimiser le Makespan ($Cmax$).

6 Les Organisations d'ateliers

6.1 Les différents types d'organisation

Les différents problèmes que l'on rencontre en ordonnancement dépendent principalement des machines et de l'enchaînement des opérations. On distingue trois catégories. La première regroupe les problèmes pour lesquels chaque tâche nécessite une seule machine (ressources).

Dans la deuxième, chaque tâche demande plusieurs machines pour son exécution. Dans la troisième catégorie, les machines sont regroupées en étages distincts.

6.1.1 Les ateliers à une ressource

Dans certains ateliers, on ne dispose que d'une ressource pour traiter un ensemble de tâches. Celle-ci ne peut effectuer le traitement à un instant donné qu'une tâche à la fois. Ce type de problème consiste à déterminer la séquence optimale de passage de n tâches sur une machine, afin d'optimiser un ou plusieurs critères donnés.

Parmi les problèmes rencontrés dans un environnement à une machine, on peut citer les problèmes de minimisation du retard maximum, du nombre de tâches en retard, ou de la somme des retards.

Malgré leur apparente simplicité, les problèmes à une machine sont NP-difficiles au sens fort [10] [11].

6.1.2 Les ateliers à plusieurs ressources

Ces ateliers sont composés d'un ensemble de m machines (ou ressources) dont chacune ne peut réaliser, à elle seule, l'ensemble des opérations. Suivant le mode de passage sur les différentes machines, trois types d'ateliers sont distingués.

- 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.

- 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.

- Open shop

C'est un atelier à cheminement libre, l'ordre des opérations n'étant pas fixé a priori; le problème d'ordonnancement consiste, d'une part, à déterminer le cheminement de chaque produit et, d'autre part, à ordonnancer les produits en tenant compte des gammes

trouvées, ces deux problèmes pouvant être résolus simultanément. Ce type d'ateliers n'est pas couramment utilisé dans les entreprises

6.1.3 Les ateliers à ressources parallèles

Dans ces ateliers, les machines sont regroupées en étages distincts. Chaque machine appartient à un seul étage et toutes les machines d'un même étage peuvent effectuer la même opération.

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

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

6.2 Relations entre organisations

La Figure1.8 illustre ces différentes relations : chaque arc de « B » vers « A » s'interprète que « A » est un cas particulier de « B ». Le job shop classique est un cas particulier de job shop flexible.

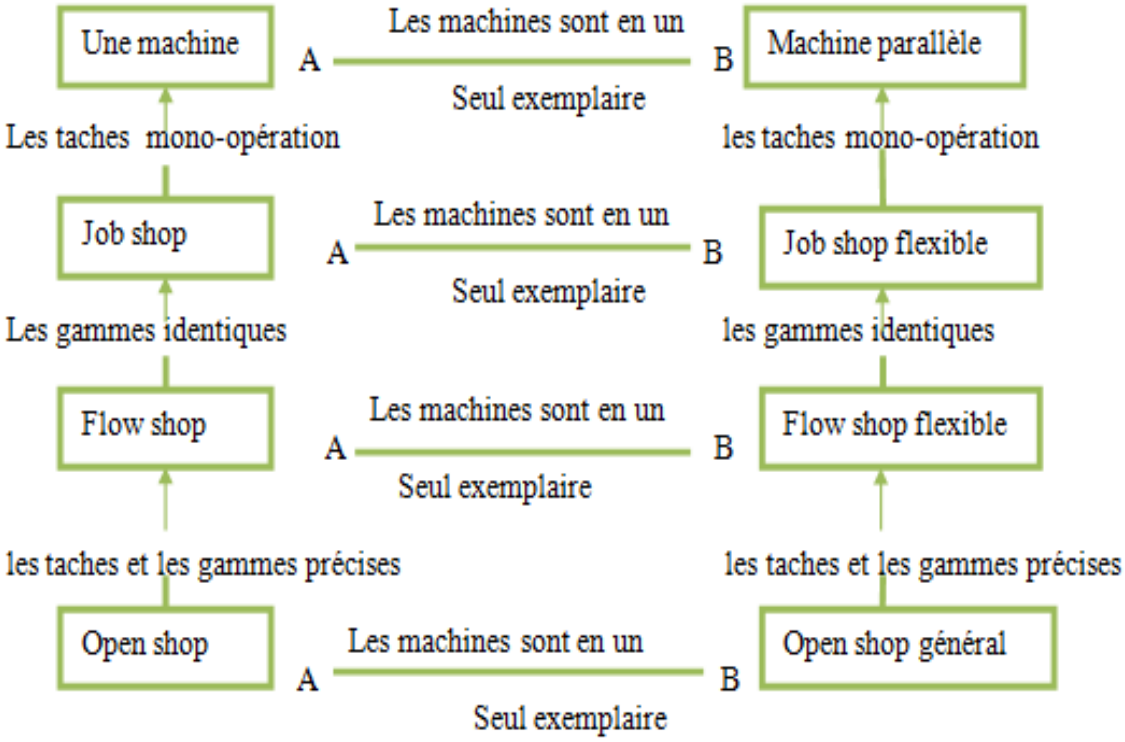

***Figure1.8*:** Relations entre les différentes organisations [12].

7 Caractéristiques générales des ordonnancements

- Ordonnancements statique et dynamique

Si l'ensemble des informations nécessaires à la résolution d'un ordonnancement est fixé au départ (ensembles des tâches, des contraintes, des ressources, etc.), on est alors devant un problème d'ordonnancement statique.

Il y a une nuance entre la solution proposée qui est, généralement accompagnée d'un plan prévisionnel d'exécution des tâches, et l'exécution réelle de ces tâches. Si le plan n'est pas respecté et les objectifs sont modifiés, on est devant un problème d'ordonnancement

dynamique qui nécessite une résolution d'une série de problèmes statiques et chaque étape doit débuter par une prise d'informations permettant d'actualiser le modèle à résoudre.

- Ordonnancements admissibles

Un ordonnancement est dit admissible s'il respecte toutes les contraintes du problème

(dates de début, dates de fin, précédence, contraintes de ressources, etc.).

- Ordonnancements actifs et semi-actifs

Dans un ordonnancement actif, aucune tâche ne peut commencer au plus tôt et qui entraîne l'ordre relatif entre au moins deux tâches.

Dans l'ordonnancement semi actif, on ne peut pas avancer une tâche sans modifier la séquence sur la ressource.

- Ordonnancements sans retard

Dans un ordonnancement sans retard, on doit exécuter la tâche qui est en attente à condition que la ressource soit disponible.

- Ordonnancement préemptif et non préemptif

Selon les problèmes, les tâches peuvent être exécutées sans interruption, c'est-à-dire si on commence l'exécution d'une tâche elle n'est pas interrompu jusqu'à son achèvement. On parle alors d'un ordonnancement préemptif. Par contre, si les tâches sont exécutées par morceaux, l'ordonnancement est appelé non préemptif.

8 Complexité des problèmes

D'une manière générale, les problèmes d'ordonnancement d'ateliers étant des problèmes combinatoires difficiles, il n'existe pas de méthodes universelles permettant de résoudre tous les cas. Plusieurs algorithmes peuvent être utilisés pour résoudre un problème d'ordonnancement mais tous ne sont pas équivalents. On peut différencier les divers algorithmes de résolution par le moyen des critères suivants :

- L'efficacité de l'algorithme en termes de durée d'exécution : un algorithme est dit plus efficace qu'un autre si pour les mêmes données, il s'exécute en un laps de temps plus court.
- L'efficacité de l'algorithme en espace mémoire de stockage : un algorithme est dit plus efficace qu'un autre si pour résoudre le même problème, il utilise moins d'espace mémoire.
- La fiabilité de l'algorithme : plus un programme est complexe, plus il y a des risques d'existence de bugs, les bugs étant des erreurs plus ou moins évidentes qui se manifestent lors de la mise en exploitation d'un programme.
- Un programme est jugé plus fiable ou plus stable qu'un autre s'il présente moins de bugs.
- La robustesse de l'algorithme : elle mesure son degré de tolérance aux erreurs des utilisateurs et sa résistance aux attaques des pirates, un programme est plus robuste qu'un autre s'il résiste mieux aux erreurs de manipulations des utilisateurs plus ou moins bien attentionnés.

Il est à noter qu'il n'y a pas de méthode ou d'échelle de mesure permettant d'évaluer la fiabilité ou la robustesse d'un algorithme. C'est à l'usage que ces qualités sont mesurées. Par contre, il existe des méthodes rationnelles et rigoureuses pour évaluer l'efficacité en temps ou en espace d'un algorithme. Ces méthodes d'évaluation portent le nom d'analyse de complexité des algorithmes.

Deux types de complexité peuvent être cités :

- La complexité méthodologique : qui exprime une fonction du nombre d'opérations élémentaires de calcul effectuées par la méthode ou par l'algorithme de résolution en fonction du nombre des données du problème traité.
- La complexité problématique : liée à la difficulté du problème à résoudre et au nombre d'opérations élémentaires qu'un algorithme déterministe peut effectuer pour trouver l'optimum en fonction de la taille du problème. Selon son degré de complexité, un problème peut appartenir à l'une des quatre classes suivantes [13] :

- Les problèmes de la classe P : dits polynomiaux s'il existe un algorithme de complexité polynomiale pour leur résolution.

- Les problèmes de la classe NP: dits problèmes NP-difficiles, qui ne peuvent à priori être résolus en un temps polynomial que par des méthodes approchées (heuristiques); au cours de leur exécution, ces algorithmes font des choix dont l'optimalité n'est pas démontrable.
- Les problèmes NP-Complets : La théorie de la NP-complétude concerne la reconnaissance des problèmes les plus durs de la classe NP. La notion de la difficulté d'un problème qui est introduite dans cette classe est celle d'une classe de problème qui sont équivalents en ce sens que si l'un d'eux est prouvé être facile alors tous les problèmes de NP le sont. Inversement, si l'un d'eux est prouvé être difficile, alors la classe NP est distincte de la classe P.

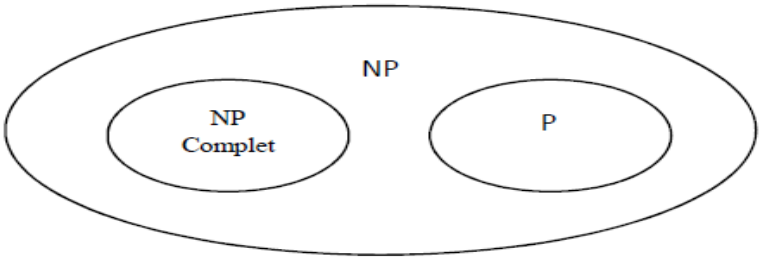

Figure1.9 : degré de complexité d'un problème d'ordonnancement.

9 Les méthodes de résolution

La plupart des problèmes d'ordonnancement sont NP-difficiles. Pour les résoudre, il existe principalement deux méthodes : les méthodes exactes (si la nature et la taille du problème le permettent) et les méthodes approchées (si le temps de calcul pour trouver une solution exacte n'est pas admissible).

- Méthodes exactes : On peut définir une méthode exacte comme une méthode qui garantit l'obtention de la solution optimale pour un problème d'optimisation. L'utilisation de ces méthodes s'avère particulièrement intéressante. Parmi ces méthodes, on peut citer les méthodes de séparation et d'évaluation, de programmation dynamique et de programmation linéaire et non-linéaire.
- La méthode par séparation et évaluation (Branch and Bound) : C'est une méthode basée sur une technique implicite énumérative. En effet, elle permet de trouver une solution optimale, en construisant un arbre de recherche, en examinant systématiquement les

sous-chemins qui sont susceptibles d'aboutir à une solution réalisable et en excluant les autres sous-arbres de la recherche [14].

- La programmation dynamique : C'est une méthode d'optimisation qui opère par phases. Son efficacité repose sur le principe d'optimalité de Bellman, à savoir « toute politique optimale est composée de sous-politiques optimales » Cette méthode permet une résolution ascendante. La solution optimale d'un problème est obtenue à partir des solutions de tous les sous problèmes.
- La programmation linéaire (PL) : C'est une des techniques classiques de la recherche opérationnelle. Cette méthode repose sur la méthode du simplexe. Elle consiste à minimiser une fonction coût en respectant des contraintes. Les critères et les contraintes sont des fonctions linéaires des variables du problème.
- La programmation non-linéaire (PNL) : Si moins une contrainte ou la fonction objective n'est pas une combinaison linéaire de variables d'optimisation, on parle alors de programmation non linéaire (PNL).
- Méthodes approchées : Les problèmes d'ordonnancement étant en général NP-difficiles, on cherche à les simplifier pour réduire le volume des calculs. Bien entendu, cela se fait au prix d'une dégradation de la qualité de la solution. Il existe plusieurs familles de méthodes qui permettent d'approcher efficacement la solution optimale. Parmi ces méthodes on trouve les méthodes heuristiques et les méta-heuristiques.

Les heuristiques sont des méthodes empiriques, qui donnent généralement de bons résultats, sans être démontrables mathématiquement. 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 (FIFO, EDD : Earliest Due Date, SPT : Shortest Processing Time,. . .) pour construire une solution proche de l'ordonnancement optimal, tout en essayant d'avoir un temps de calcul raisonnable .Les méta-heuristiques représentent des concepts généraux de résolution. Il faut formuler le problème abordé de telle manière qu'il soit adapté à l'application de ces concepts. Parmi les méthodes méta-heuristiques on a :

- les méthodes par construction.
- les méthodes par décomposition.
- les méthodes par voisinage (recherche tabou, algorithmes génétiques,. ..).

CHAPITRE 2

DESCRIPTION DU PROBLEME D'ORDONNANCEMENT JOB SHOP

1 Introduction

Dans ce chapitre, nous présentons les problèmes d'atelier à cheminement multiple, communément appelé sous le terme anglais « job shop ». Ces problèmes sont les plus généraux parmi ceux de la famille de l'ordonnancement.

le problème job shop est l'un des problèmes d'ordonnancement combinatoire les plus connus. la résolution du problème est un tâche difficile, le classant ainsi parmi les problèmes NP-Difficiles au sens fort , dont la fonction objectif est la minimisation du Makespan
Ce chapitre est composé de cinq parties..la première partie présente le problème job shop général.la deuxième partie présente les types de job shop .la troisième partie présente modélisation graphique .la quatrième partie présente la complexité du job shop.la dernière et cinquième partie présente les méthodes de résolution du problème de job shop.

2 Description du problème d'ordonnancement job shop

2.1 Historique du Job-Shop

L'histoire du Job-Shop a commencé il y a plus de quarante ans. Cependant c'est dans les années 1960 qu'est apparu un problème d'ordonnancement aujourd'hui bien connu, celui de Fisher et Thompson avec 10 jobs et 10 machines Fisher et Thompson (1963). Ce problème résista pendant près de 25 ans et engendra une compétition entre les chercheurs du domaine Blazewicz et al., (1996). Dans cet article les auteurs dressent un état de l'art du JS et donnent une définition formelle du problème. Ils décrivent également les méthodes de résolution à l'aide d'équations linéaires ainsi qu'une modélisation sous forme d'un graphe disjonctif-conjonctif. Les méthodes de résolution sont séparées en deux familles, à savoir :

- les méthodes exactes pour lesquelles ils donnent tous les éléments nécessaires la construction et l'exécution d'un algorithme exact et les méthodes de branchement et les inégalités valides .

- les méthodes approchées pour les quelles ils décrivent quelques règles de priorité, la procédure à machine goulot, les algorithmes de recherche locale ainsi que des procédures d'ordonnancement opportunistes. Ils énoncent également les principaux voisinages utilisés dans la littérature.

Dans Jain et Meerran (1999) on trouve, un bon état de l'art concernant le job-shop de son début jusqu'à la fin des années 90. Il liste les articles en les classant par méthode de résolution utilisée pour le JS. En plus de la description de la méthode, Jain et Meerran proposent une liste complète d'instances. Cette dernière sert de base de comparaison entre les méthodes de résolution. Ils présentent également des tableaux de synthèse dans lesquelles ils donnent une vue d'ensemble récapitulative des méthodes et des extensions ainsi que les références les évoquant. Les méthodes exactes commencent avec les premiers travaux menés par Manne (1960), puis par Brooks et White (1965) suivi en 1968 par Greenberg qui utilisa une formalisation linéaire en nombre entiers Greenberg (1968). S'en suivi alors de nombreuses publications notamment celles de Ficher dans Fisher (1973a) et Fisher (1973b) qui utilisa les multiplicateurs de Lagrange. Dès les années 1975 McMahon et Florian développent un algorithme McMahon et Florian (1975) dépassant la performance des algorithmes de Fisher. Durant les vingt dernières années furent créés des algorithmes plus performants utilisant entre autre la relaxation lagrangienne qui consiste en l'abandon de certaines contraintes pour résoudre des problèmes plus proches de la réalité avec plus d'une centaine de jobs et d'une cinquantaine de machines. Cette technique de relaxation fut travaillée afin d'associer à chaque contrainte relâchée une pénalité appelée technique de relaxation lagrangienne augmenté. Plus tard, en 1989, Carlier et Pinson (1989) décrivent un algorithme basé sur le Branch and Bound, qui pour la première fois, résout de manière optimale le problème 10x10 de Fisher et Thompson ainsi que d'autres plus complexes. Plus récemment nous pouvons retenir les approches proposées dans Caseau et Laburthe (1995), Perregaard et Clausen (1995) et Blazewicz et al., (1998). Cependant bien que ces méthodes de résolutions des problèmes de type Job-Shop fournissent des résultats exacts, elles nécessitent souvent un temps d'exécution très long pour des problèmes de type FT 10x10. C'est pour cela qu'en parallèle de ces approches, d'autres, pouvant être qualifiées de méthodes approchées, ont été développées.

Les méthodes approchées offrent une alternative intéressante, car elles permettent d'obtenir des résultats rapidement. Ces dernières peuvent être envisagées pour le traitement de problèmes réels (généralement de grande taille). Comme il est très difficile d'établir un état de l'art exhaustif d'apparition des méthodes approchées pour le job-shop, on a choisi l'ordre chronologique. Les premières méthodes sont donc des heuristiques de construction apparues à la fin des années 50 et début des années 60 comme celles proposées par Giffler et Thompson (1960) et Fisher et Thompson (1963). Dans les années 70, malgré cet intérêt réel pour les méthodes exactes, on note la naissance des méthodes de relaxations, suivi dans les années 80, de l'apparition des Méta-heuristiques telles que les algorithmes génétiques, la recherche Tabou et le recuit simulé. Ensuite, on a vu l'émergence de méthodes telles que la programmation par contraintes, les réseaux de neurones, les systèmes expert ainsi que le recours à la simulation, afin de résoudre les problèmes d'ordonnancement.

2.2 Présentation du problème job shop

Le problème Job-Shop est un des problèmes fameux d'optimisation combinatoires NP-Difficiles [15]. Il peut être expliqué comme suit: Étant donné un ensemble de n jobs et un ensemble de m machines. Chaque job j est composé d'un ensemble ordonné de m opérations O1, O2, O3,…, On. L'ordre des opérations ne peut pas être modifié, elles sont organisées par un ordre donné propre à chaque job. En outre, l'opération Oi doit être réalisée par une machine donnée durant un temps Ti sans interruption (i.e. lorsque l'opération commence, elle ne peut pas être interrompue ou stoppée temporairement jusqu'à ce qu'elle s'achève complètement). D'autre part, chaque machine peut exécuter un seul job à la fois (i.e. dans une période donnée) et chaque job peut être exécuté par une seule machine pendant une période de temps. Le problème consiste à trouver une allocation faisable (ou l'allocation optimale) de toutes les opérations à des intervalles de temps et des machines disponibles en respectant les contraintes du problème. L'objectif final est de minimiser le temps d'exécution du dernier job. Il existe des variantes du problème Job-Shop comme celle nommée: Multi Purpose Machine Job-Shop .le job shop se caractérise par un cheminement multiple, puisque les opérations de chaque job peuvent emprunter divers chemins (routage des opérations).

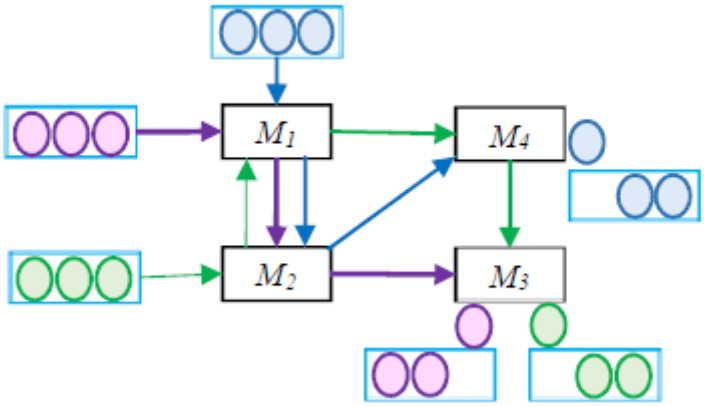

***Figure 2.10* :** problème de job shop compose 3 jobs et 4 machines.

Généralement on distingue deux organisations d'ateliers job shop

- job shop simple : s'il existe pour chaque machines un seul exemplaire.
- job shop hybride : s'il existe au moins une machine disposant de plusieurs exemplaires.

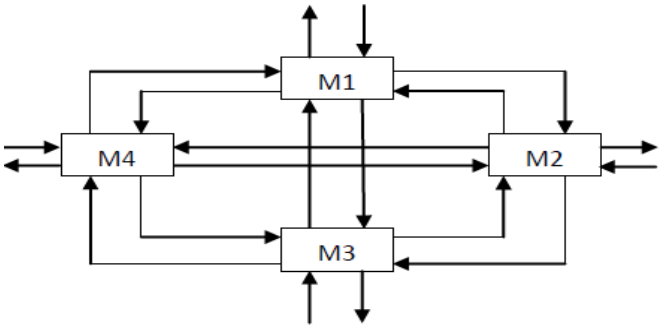

Figure 2.11:job shop simple avec 4 machines.

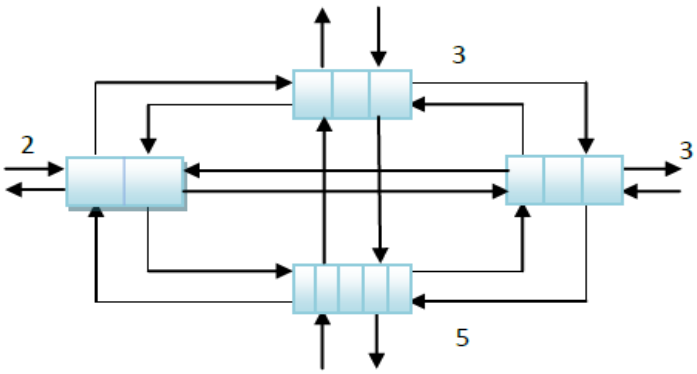

***Figure 2.12*:** job shop simple & hydride.

2.2.1Makespan ($\min_{1\leq i\leq n}(C\max_i)$)

Le Makespan est un critère d'optimisation, représente la date de fin d'exécution de la dernière

opération qui quitte l'atelier. Il est particulièrement intéressant puisque, il permet de déterminer les séquences critiques, c'est-a-dire les séquences sur lesquelles tout retard a des conséquences sur le temps de réalisation des jobs. Sa minimisation implique la minimisation de la durée de séjours des produits dans l'atelier, des encours ainsi que le temps non productif des machines.

Makespan est la date de fin du dernier job qui sort de l'atelier de production. Il est défini par l'équation :

$$f(1) = \min_{1\leq i\leq n}\left(\max_{1\leq j\leq ni}(Cij)\right)$$

2.2.2 les contraintes

Ces contraintes touchantes a la fois les possibilités d'utilisation des machines et les liens qui peuvent exister entre les opérations

En outre, les contraintes différent d'une formulation a l'autre (selon le type du job shop) mais ne retiendrons ici que le cas de job shop simple ,dont les contraintes sont les suivantes :

- les machines sont indépendantes les unes des autres (pas d'utilisation d'outil commun, par exemple).
- Les taches sont indépendantes les unes des autres en particulier, il n'existe aucun ordre de priorité attache aux taches.
- Une tache ne peut être en état d'exécution que sur une seul machine à la fois .deux opérations de la même tache ne peuvent être exécutée simultanément.
- Une machine ne peut exécuter qu'une seul opération à une instante donne.
- Seulement le temps d'exécution proprement dit, est pris en compte .le temps de transport d'une machine a l'autre, de préparation, etc.ne sont pas considères.
- Les machines sont disponibles jusqu'à a la fin de l'ordonnancement .on particulier, les pannes de machines ne sont pas prises en compte.

- La définition des taches est déterminée avant lancement de l'ordonnancement, i.e. il n'existe pas d'événements aléatoires pendant l'exécution.

2.2.3 les objectifs

L'objectif du problème d'ordonnancement est de fixer les dates de début des opérations pour cela, il faut déterminer l'ordre de passage de l'ensemble des taches sur chaque machine , en respectant les contraintes du problème . le but est ensuite, de minimiser ou maximiser une fonction objectif, pour trouver la ou les meilleur(s) solution(s).

Dans notre cas , et pour les problèmes traites, nous prendrons comme objectif, la durée total de l'ordonnancement **Cmax**, puisque d'un cote ,c'est un critère simple et le plus utilisée dans la littérature ;et dans autre coté , notre travail n'exige pas des critères trop compliques ,un critère simple suffit à la réalisation de l'étude.

Nous pouvons proposer la fonction objectif de JSSP, qui est :

Minimiser Cmax = C n×m

ce qui mène à conclure que la fonction objectif est de minimiser le Makespan Cmax.

2.3 Types du problème job shop

2.3.1 Problème Job shop classique (JSC)

Le problème d'ordonnancement de job shop constitue l'un des problèmes d'ordonnancement d'atelier les plus étudié dans la littérature, les variations autour du problème de job shop sont nombreuses et il n'est pas possible de trouver dans la littérature une formulation unique de celui-ci. Nous présentons ici la formulation la plus générale possible du problème de job shop simple.

Le problème d'ordonnancement de type job shop consiste à réaliser un ensemble de n jobs sur un ensemble de m machines. En cherchent d'attendre certains objectifs. Chaque job Ji est composé d'une suite de ni opérations devant être exécutées sur les différentes machines selon un ordre préalablement défini. Par ailleurs, un ensemble de contraintes concernant les machines et les tâches souvent existées, comme :

- Les machines sont indépendantes les unes des autres.

- Une machine ne peut exécuter qu'une seule opération à la fois à un instant donné.
- Les machines sont disponibles pendant toutes la durée de l'ordonnancement.
- La capacité de stockage entre les machines est considérée comme infinie.
- La préemption des opérations n'est pas autorisée.
- L'ordre de passage des opérations d'un job sur les différentes machines est fixe pour chaque job et peut être différent d'un job à un autre [16].

Figure 2.13 représente un problème type de job shop classique composé de 3 jobs et 5 machines, les gammes opératoires sont les suivantes:

J1 : M1, M2, M3, M4, M5.

J2 : M1, M5, M4.

J3 : M2, M3, M4.

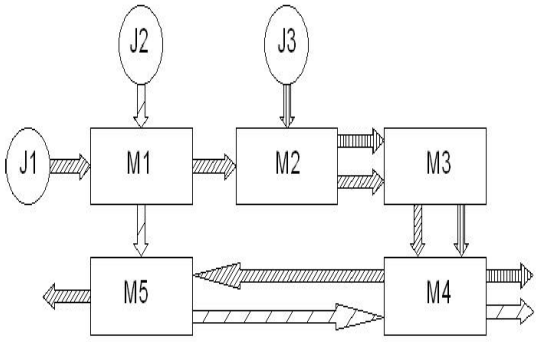

Figure 2.13 : problème de job shop classique compose 3 jobs et 5 machines.

2.3.2 Problème Job shop flexible (FJSP)

Le job shop flexible est une extension du modèle job shop classique. Sa particularité essentielle réside dans le fait que chaque opération peut être exécutée sur plusieurs machines. Dans ce modèle, les machines qui effectuent la même opération sont groupées dans un même étage. Il s'ensuit qu'il offre plus de flexibilité par rapport au job shop classique grâce à la polyvalence de ces machines. Toutefois, cela induit une complexité supplémentaire due à la

nécessité de la détermination des affectations adéquates avant d'établir l'ordre de passage des différentes opérations sur les machines.

Une remarque importante dans les problèmes de type job-shop hybride est que le nombre de machines peut varier d'un étage à l'autre, ainsi que les performances des machines qui ne sont pas forcément les mêmes pour toutes les machines d'un même étage. Ces performances nous permettent de classer les systèmes avec machines parallèles en trois groupes, à savoir :

- Machines identiques : La durée d'exécution d'une tâche est la même sur toutes les machines.
- Machines uniformes : La durée d'exécution d'une tâche est similaire sur toutes les machines.
- Machines indépendantes : La durée d'exécution d'une opération dépend de la machine sur laquelle elle est exécutée

2.4 Modélisation graphique du problème job-shop

La modélisation, est en général, une étape très importante dans la résolution d'un problème d'ordonnancement. C'est une écriture simplifiée de toutes les données du problème permettant d'en traduire tous les détails pour mieux représenter la réalité des choses.

Notation :

Dans ce qui suit nous allons donner les notations des données utilisées dans le cadre de notre problème [16] :

M = {M1, M2,…, Mm} : Ensemble de toute les machines disponibles

$\boldsymbol{n}$: Nombre totale de jobs

m: Nombre totale de machines

$\boldsymbol{i}$: indice du $\boldsymbol{ieme}$ job

$\boldsymbol{j}$: Indice de la $\boldsymbol{j}$é$\boldsymbol{me}$ opération du job $\boldsymbol{ji}$

$\boldsymbol{Oij}$: $\boldsymbol{j}$é$\boldsymbol{me}$ Opération du job $\boldsymbol{ji}$

Mij : Machine sur laquelle l'opération Oij est exécutée

Pij: Durée d'exécution de l'opération Oij

Sij: Date de début de l'opération Oij.

Cij : Date de fin de l'opération Oij.

Cmax=max {Ci, i = 1,…,n} : Makespan ou date de fin de tous les jobs

il existe plusieurs manières de modéliser graphique du problème de job shop Le job shop peut être modélisé de différentes façons :

2.4.1 le graphe disjonctif

Le graphe disjonctif a été proposé par Roy et Susmann en 1964 pour l'ordonnancement des job-shop.

Le graphe disjonctif est composé de sommets qui représentent les différentes opérations des jobs et de deux sommets pour indiquer le début (source) et la fin des jobs. Les sommets sont reliés entre eux par deux types d'arcs : arc conjonctif et arc disjonctif.

Les arcs conjonctifs relient deux opérations consécutives d'un même job, décrivant ainsi la contrainte de précédence entre les opérations. Les sources sont connectées aux premières opérations de chaque job par un arc conjonctif.

Les arcs disjonctifs relient les opérations qui s'exécutent sur la même machine et qui n'appartiennent pas au même job

La solution réalisable est obtenue en orientant les arcs disjonctifs de façon à avoir un graphe acyclique (sans circuit). Le graphe disjonctif acyclique est appelé un graphe arbitré [2]**.**

La formalisation du problème job shop par graphe disjonctif est parmi les formalisations les plus utilisées dans la littérature.

Un problème de job shop peut alors se modéliser par un graphe disjonctif peut être représenté par G = (X, C,D) ou :

➢ **X** est l'ensemble des nœuds correspondant à toutes les opérations avec deux nœuds

Supplémentaires, un nœuds source **S** et puits **T**.

➢ **C** est l'ensemble des arcs conjonctifs qui représentent les contraintes de précédence entre les opérations d'un seul job.

➢ **D** est l'ensemble des arcs disjonctifs qui connectent deux opérations procédant sur la même Machine.

La figure :correspond au graphe disjonctif d'un atelier composé de 3jobs et 3 machines ,tels que :

$X=\{O_{1.1}, O_{1.2}, O_{1.3}, O_{2.1}, O_{2.2}, O_{2.3}, O_{3.1}, O_{3.2}, O_{3.3}\}$

- 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.
- Les arcs conjonctifs représentent les contraintes de précédence entre les opérations d'un même job (exemple : $O_{1.1}$, $O_{1.2}$ et $O_{1.3}$)
- La clique cyclique entre les sommet $O_{1.1}$, $O_{1.2}$ et $O_{1.3}$ signifie un conflit d'utilisation d'une même ressource.

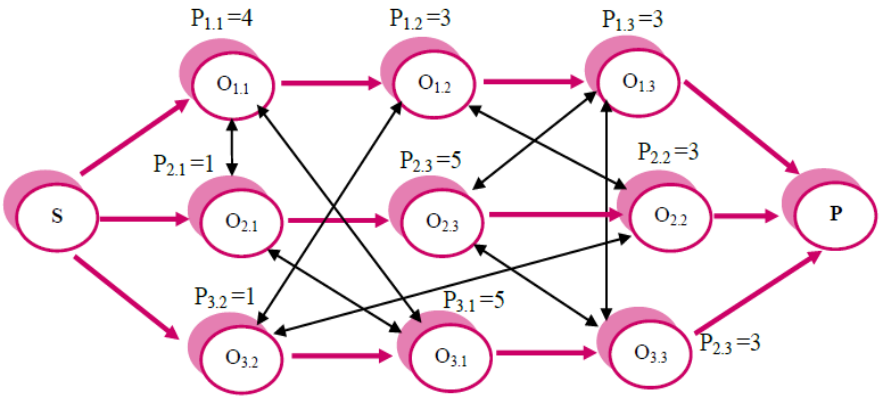

***Figure 2.14* :**le graphe disjonctif d'un problème de job shop (3*3).

2.4.2 le diagramme de Gantt

qui associe à chaque opération une barre horizontale avec une longueur proportionnelle au temps de l'opération.

Comme montre dans la figure 2.3, il existe deux types du diagramme de Gantt. Diagramme de machine (figure 2.3(A)) qui associe les barres aux machine et le diagramme de job (figure 2.15 (B)) associant les barres aux opérations des jobs [7][17].

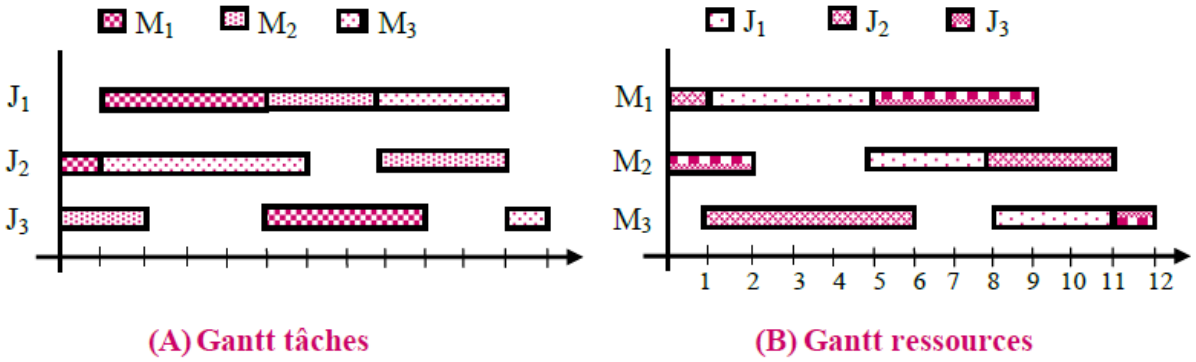

Figure 2.15 : diagramme de Gant (problème de job shop (3*3)).

2.5 Complexité des problèmes job-shop

Les problèmes de job shop sont en général NP-complets, même si l'atelier est simple. En effet, [10] ont montre que les ateliers possédant plus de trois machines, ou un nombre de taches supérieur ou égal a trois, sont NP-difficiles même si la préemption est permise.

De même, pour les problèmes a deux machines, des qu'il y a recirculation, ils deviennent fortement NP-difficiles. L'expérience a montre que les problèmes de job shop ayant un nombre m>2 machines, et optimisant les critères C max et F sont NP-difficile au sens fort , même en utilisant des heuristiques. Théoriquement, le problème du job shop J//C max peut se réduire polynomialment aux problèmes J//T max et J//L max. De même, le problème J//F peut se réduire aux problèmes J//T et J//L. par conséquent, ces quatre dernières instances du job shop sont NP-difficile .Pour résoudre un problème d'ordonnancement de manière efficace, il faut prendre en compte la particularité de chaque problème. C'est pourquoi il faut s'assurer de la classe de complexité associe au problème a ordonnancer. Il existe plusieurs méthodes de résolution du problème d'ordonnancement, la connaissance de la classe de complexité peut nous aider énormément quant au choix judicieux de la méthode de résolution.

2.6 Méthodes de résolution du problème de job shop

Nous avons vu précédemment que le problème d'ordonnancement de job shop à l'exception de certains cas, est NP-difficile.les problèmes appartenant à la classe P ont des algorithmes polynomiaux permettant de les résoudre .pour les problèmes appartenant à la classe NP ,l'existence d'algorithmes polynomiaux semble peu realiste.differentes méthodes de résolution (exactes ou heuristiques),sont largement utilisées pour appréhender les problèmes de job shop NP-difficiles.

2.6.1 les différentes méthodes de résolution

Une présentation de la totalité des méthodes et de leurs variantes apparait une tache difficile au vu du nombre d'approches proposées. Nous présentons sur le tableau les grandes orientations des méthodes, en essayant de repérer les travaux de recherche essentiels. Ce tableau est synthétisé à partir des travaux de [18].pour les méta-heuristiques , nous essayons de circonscrire avec plus de détails – sans prétendre toutefois être exhaustif – l'important des travaux effectués sur ces méthodes dans un paragraphe à part.

Méthodes de résolution			Principaux auteurs
Exactes	Méthodes pour les problèmes polynomiaux	Le problème à deux machines	Johnson(54),Akers(56) Jackson(56),Szwarc(60)
		Le problème à deux tâches	Akers (56),Brucker(88.94),Brucker & Jurisch(93) Kravchenko & Sotskov(96),Kubiak & Timkovsky(96),Timkovsky(97),williamson & al.(97),brucker & al. (97).
	La programmation mathématique		Bowman (59),wagner(59) ,Manne(60),Balas(65),dyer & Wolsey(90),Van Den Akker(94).
	Le « branch and bound »		Cook(71).lageweg & al .(77à Prinson(88,95).
	Les methodes de relaxation		Fisher(73,76) Fisher & al.(75).Van de Velde (91),Hoitomt & al .(93).Della Croce & al.(93) Hoogeveen & Van De Velde(95).
	Les methodes de décomposition		Ashour (67) kanzedal (83),Chu et al.(92) Kruger et al.(95).
	Les heuristiques constructioves	L'approche par opérations	Rowe & Jackson(56).Giffler & Thompson(60).Fisher & Thompson(63), Crowston et al.(63).Viviers(83).Sabuncuoglu & Bayiz(97).

approximatives	Les méta-heuristiques	Algorithmes génétiques	Davis(85).Falkenauer & Bouffouix(91).Nakano & Yamada(91,92).Kobayashi & al.(95).Cheng & al.(96).shi(97).
		Recuit Simule	Matsuo & al.(88).Kolonko(98).Aarts & al.(91,94).
		Recherche Tabou	Taillard(89,94),Thomesn (97).Jain & al. (98).

***Tableau 2.1* :**Principales méthodes de résolution problème de job shop.

CHAPITRE 3

ADAPTATION DE LA METHODE COUCOU AU PROBLEME JOB SHOP

1 Introduction

Le coucou est un oiseau discret, longiligne, de taille moyenne et son chant marque le début de la belle saison. Parmi ses caractéristiques fascinantes, on cite aussi sa stratégie de reproduction. Le parasitisme des couvées chez un certain nombre d'espèces, est le plus étudie et discuté. Les coucous femelles mettent un ou plusieurs œufs dans des nids, des autres espèces d'oiseaux, précédemment observées. Le but est de garantir un passage à la génération suivante en laissant les oiseaux hôtes guides par leur instinct naturel d'éclosion, de couvaison et d'apporter la nourriture aux petits coucous. An d'augmenter la probabilité d'avoir un nouveau coucou, la femelle gobe un œuf dans le nid parasite, avant d'y pondre le sien. Certaines espèces hôtes peuvent avoir un conflit avec le coucou intrus. Quand des oiseaux hôtes découvrent la présence d'œuf ne leur appartenant pas, grâce à une aire de peau sensible et dénudée qu'ils ont alors sous le ventre, soit ils s'en débarrassent, soit ils abandonnent le nid en construisant un nouveau ailleurs. Certains types de coucous comme le Tapera a développé la capacité d'imiter les œufs d'un ensemble d'hôtes en couleur et en forme.
Dans ce chapitre, nous allons présenter une description de l'algorithme que nous avons pris comme un point de départ vers la résolution des problèmes d'optimisation combinatoire tels que JSSP. Nous allons commencer par une description de Cuckoo Search (CS).

Figure 3.16 **:** Oiseau de coucou.

2. Le comportement et la reproduction des Coucous

Toutes les espèces d'oiseaux ont la même approche de prolifération. Aucun oiseau ne donne naissance à des petits vivants. Après l'accouplement, l'oiseau pond un œuf recouvert d'une coquille protectrice qui est ensuite incubée en dehors du corps. Vu la richesse en protéine des œufs d'oiseau, ils construisent un volumineux repas pour toutes sortes de prédateurs. Afin de protéger leurs œufs des menaces des différents prédateurs, les oiseaux doivent trouver un endroit sûr pour donner plus de chance d'éclosion à leurs poussins. Trouver un endroit pour placer en toute sécurité, couver leurs oeufs et élever leurs poussins au point de l'indépendance, est un défi pour les oiseaux. Ces derniers l'ont résolu de nombreuses manières astucieuses en utilisant des conceptions complexes et artistiques. De nombreux oiseaux construisent des nids isolés, cachés dans la végétation pour échapper au risque d'être détecté par les prédateurs. Certains d'entre eux sont des spécialistes dans le choix du bon emplacement de leurs nids. Ils cachent bien leurs nids de façon à ce que même les yeux de l'homme qui voit tout n'aient pratiquement jamais pu les détecter .Il ya un autre type d'oiseaux qui se dispensent des responsabilités imposées par leur lien de parentalité. Ce sont des oiseaux parasites. Ils ne construisent pas leurs propres nids. En effet, ils pondent leurs propres oeufs dans les nids des autres espèces en confiant la responsabilité d'incubation, de nourriture et d'élevage de leurs oeufs aux parents adoptifs.

Les coucous sont un bon exemple d'oiseaux parasites de couvée. Ce sont des oiseaux fascinants, ils ont un bon chant. Ils appartiennent à la classe des cuculidés. Cette dernière compte environ 140 espèces réparties partout dans le monde. Plusieurs espèces de coucous sont des parasites. Ils ne construisent pas leur propre nid pour pondre leurs oeufs. Quelques espèces entre eux s'engagent à pondre leurs oeufs dans des nids communautaires où plus d'une femelle pond dans le même nid, comme l'Ani et le Guira [19]. En fait, ils pondent leurs oeufs dans des grands nids collectifs des membres de leur propre espèce puis ils se chargent de l'élevage collectif de leurs petits poussins. D'autres espèces pondent leurs oeufs dans des nids d'autres espèces [20] puis ils se sauvent, comme le coucou gris, le coucou koël, Le Coucou plaintif et le coucou geai. Les coucous de ce dernier type ne couvent pas et ne nourrissent jamais leurs poussins. Quelques explications tentent de traduire ce comportement par le fait que le coucou parent se nourrie d'insectes toxiques. Cette nourriture menace la vie de ses poussins ce qui pousse le coucou parent à confier la responsabilité de nourriture et

d'élevage de ses petits à d'autres espèces capables de leur fournir la bonne et la saine nourriture.

De nombreuses espèces d'oiseaux apprennent à reconnaître les oeufs coucous déversés dans leurs propres nids. Dans le cas ou l'oiseau hôte arrive à découvrir l'oeuf coucou dans son nid, il va soit jeter l'oeuf étranger hors son nid ou abandonner son propre nid. Tandis que les hôtes tentent de trouver des moyens de détection de l'oeuf parasite, le coucou cherche constamment à améliorer le mimétisme du modèle des oeufs de ses hôtes [21]. Le coucou est un expert dans l'art de tromperie. Sa stratégie se base sur la furtivité, la surprise et la vitesse [21]. Après l'accouplement, le coucou pond ses oeufs dans les nids des autres espèces. L'oeuf coucou risque de ne pas survivre au cas où l'oiseau hôte découvre qu'il n'est pas le sien. Dans le but d'augmenter la ressemblance entre l'oeuf coucou et les oeufs d'accueil, certaines femelles coucou sont très spécialisées dans le mimétisme de couleurs et de modèles des oeufs de l'hôte. Chaque femelle coucou se spécialise sur une espèce hôte particulière. En outre, Les mamans coucous peuvent détruire les oeufs de l'oiseau hôte afin d'hausser la possibilité d'éclosion et de nourriture de leurs oeufs.

En général, les oeufs du coucou se développent plus rapidement et éclosent plus tôt que ceux de l'oiseau d'accueil. Dès son éclosion, le poussin coucou dupe ses parents adoptifs. Il éjecte les oeufs d'accueil par aveuglement ce qui lui offre la monopolisation des soins de ses parents adoptifs et de la nourriture destinée à toute une couvée. Il incite ses parents adoptifs à suivre le rythme de son taux de croissance élevé par son appel de mendicité rapide [22] qui imite l'appel des poussins d'accueil et aussi sa bouche ouverte qui construit un fort stimulus [23]. En effet, il grandit plus vite au détriment des poussins d'accueil.

2.1 L'habitat des coucous

L'habitat est le milieu de vie, de reproduction et de développement d'une population d'une espèce donnée. De même pour les coucous, il constitue une source de nourriture et un lieu de reproduction. Les Coucous se produisent dans une grande variété d'habitats. La majorité des espèces survivent dans les forêts et les bois, principalement dans les forêts tropicales à feuilles persistantes. En plus des forêts, certaines espèces de coucous occupent des environnements plus ouverts, ce qui peut inclure même les zones arides comme les déserts [24]. A titre d'exemple, le Coucou plaintif fréquente une assez grande

variété d'habitats tels que les bois ouverts, les forêts secondaires, arbustes et broussailles, les champs cultivés et aussi les jardins, aussi bien en milieu rural que citadin. On peut aussi le voir dans les herbages et les marais. Le Coucou gris fréquente les forêts de conifères ou de feuillus, et les zones boisées en général, les espaces boisés ouverts, les lisières de forêts et les clairières, les steppes arborées, les prairies, les marais et les roselières, ainsi que les zones cultivées avec des arbres et des buissons. Le Coucou geai fréquente des habitats semi-arides tels que les zones boisées ouvertes (surtout avec des acacias et des arbustes épineux), les contreforts rocheux des collines dans les savanes sèches, et les zones agricoles sèches avec des arbres et des buissons. On le voit souvent voler au-dessus des espaces découverts. Selon la distribution, et particulièrement en Europe, on peut le trouver dans les landes de bruyères avec du chêne-liège et des conifères du genre Pinus pinea. Il fréquente également les bosquets et les plantations d'oliviers [19].

2.2 La migration des coucous

La migration est un phénomène très répandu chez certains animaux y compris les oiseaux. Ce phénomène n'est pas obligatoire. En fait, il est lié à des paramètres alimentaires, physiologiques et hormonaux. En fonction de ces paramètres, l'oiseau est incité à se déplacer ou non. La date de migration n'est pas commune. Certaines espèces partent avant de manquer de nourriture, d'autres attendent quelques changements climatiques. En effet, l'oiseau risque de perdre sa vie en cas de baisse de disponibilité de nourriture ou des conditions climatiques rigoureuses. La plupart des espèces de coucous sont sédentaires, mais plusieurs espèces de coucou entreprennent des migrations saisonnières, et plusieurs autres entreprennent des migrations partielles sur une partie de leur distribution. La migration peut être diurne, comme celle du coucou 'Channel-billed' comme elle peut être nocturne, comme celle du coucou à bec noir et du coucou à bec jaune qui se reproduisent en Amérique du nord et migrent de nuit à travers le Mexique et l'Amérique centrale pour hiverner en Amérique du Sud. En général, les coucous parasites migrent sur de longues distances et ont un vol rapide et direct. Leurs déplacements coïncident non seulement avec la saison de reproduction des espèces hôtes, mais aussi avec l'émergence des chenilles qui constituent leur nourriture favorite [19]. Une preuve tangible a été fournie par un coucou bagué en 1939, à sa naissance, et repris en 1952, en Allemagne. Cet oiseau avait effectué 26 trajets entre l'Europe et l'Afrique, soit au minimum 150 000 km. L'équivalent de presque quatre fois le tour du globe.

3 Description de la recherche du coucou (CS)

La recherche de coucou (CS) est basée sur la population inspirée par la nature Algorithme méta heuristique initialement proposé par Yang et Deb en 2009 pour résoudre les problèmes d'optimisation .Le algorithme est inspiré par la couvée interspécifique obligatoire parasitisme de certaines espèces de coucous qui pondent leurs œufs dans le des nids d'oiseaux hôtes d'autres espèces dans le but de s'échapper de l'investissement parental dans l'élevage de leur progéniture. Ce stratégie est également utile pour minimiser le risque de perte d'œuf à d'autres espèces, car les coucous peuvent distribuer leurs œufs parmi nombre de nids différents. Bien sûr, parfois ça arrive que les oiseaux hôtes découvrent les œufs exotiques dans leurs nids.

Dans de tels cas, l'oiseau hôte peut prendre différentes réponses actions allant de jeter de tels œufs à simplement quitter le nid et en construire un nouveau ailleurs. cependant, les parasites de la couvée ont à leur tour développé sophistiqué stratégies (telles que des périodes d'incubation plus courtes, des Croissance au nid et coloration de l'œuf ou motif mimant leurs hôtes) pour veiller à ce que les oiseaux hôtes prennent soin de la les oisillons de leurs parasites. Ce modèle de comportement d'élevage intéressant et surprenant est la métaphore de la méta heuristique de recherche de coucou approche pour résoudre les problèmes d'optimisation. Dans le coucou algorithme de recherche, les œufs dans le nid sont interprétés comme une piscine des solutions candidates à un problème d'optimisation, tandis que l'oeuf de coucou représente une nouvelle solution à venir. L'ultime but de la méthode est d'utiliser ces nouvelles (et potentiellement mieux) solutions associées aux œufs de coucou parasite pour remplacer la solution actuelle associée aux œufs dans le nid. remplacement, effectué itérativement, conduira éventuellement à un très bonne solution du problème.

3.1 composantes principales du recherche coucou (CS)

3.1.1 nid

Dans CS, un nid jouit des caractéristiques suivantes :

- Le nombre de nids est fixé
- Un nid est un individu de la population.
- Le nombre de nids est égal à la taille de la population.

- Un nid abandonné implique le remplacement d'un individu de la population par Un nouveau.
- Un œuf dans un nid représenté une solution.

Donc, on peut conclure de ces caractéristiques que dans le cas de notre problème

D'optimisation combinatoire, un nid se figure comme un individu de la population avec sa propre solution.

3.1.2 œuf

Un Coucou peut pondre un seul œuf dans un seul nid, ce qui donne aux œufs les

Propriétés suivantes :

- Un œuf dans un nid est une solution adoptée par un individu de la population.
- Un œuf du coucou est une nouvelle solution candidate pour une place dans la Population.

3.1.3 fonction objectif

La fonction objectif (de test, fitness) est une fonction qui, à chaque solution dans L'espace de recherche associe une valeur numérique pour décrire sa qualité ou fitness. Donc la qualité ou fitness d'une solution est proportionnelle à la valeur de la fonction Objectif. Dans notre problème, un nid de meilleure qualité d'œuf nous mènera vers les nouvelles générations. Ce qui veut dire que la qualité d'un œuf de Coucou est directement liée à sa capacité de donner un nouveau Coucou.

3.2 Le principe de la recherche coucou

La recherche coucou est un méta heuristique très récente, proposé par Xin-She et Deb

en 2009. En s'inspirant du comportement des coucous dans leur reproduction, Yang et Deb se Sont basés sur trois principes pour proposer leur nouvelle méta heuristique

- Chaque coucou pond un seul œuf à la fois. Il le dépose dans un nid qu'il choisit Aléatoirement.

- Les meilleurs nids qui incluent des œufs (solutions) de bonnes qualités vont être les élus qui construisent les membres de la nouvelle génération.
- Le nombre des nids hôtes valides est fixé. L'oiseau hôte peut détecte le coucou Étranger avec une probabilité $Pa \in [0,1]$ Dans ce cas-là, l'oiseau hôte tranche entre Écarte le coucou de son nid en lui éjecter hors nid ou abandonner son nid pour aller Construire un autre dans une nouvelle position.

3.3 Algorithme de la recherche coucou

Entrée : Population de nid $\boldsymbol{xi}$=($\boldsymbol{xi1}$,…….$\boldsymbol{xiD}$)$\boldsymbol{T}$pour i= 1,…….$\boldsymbol{Np}$

Sortie : Meilleur solution ($\boldsymbol{xbest}$)

1:Générer population initiale des nids hôtes ()

2 : eval=0 ;

3 :**tant que** (condition arrêt non satisfaite)**faire**

4 :**pour** i-1 **:$\boldsymbol{Np}$faire**

5 :$\boldsymbol{xi}$= nouvelle solution de génération ($\boldsymbol{xi}$) & correspondant à la valeur $\boldsymbol{fmin}$=$\boldsymbol{min}$($\boldsymbol{f}$($\boldsymbol{x}$))

6 :$\boldsymbol{fi}$=évaluation de nouvelle solution ($\boldsymbol{xi}$)

7 : eval=eval+1 ;

8 :j=[rand (0,1)*$\boldsymbol{Np}$+$\boldsymbol{1}$] ;

9 :**Si**$\boldsymbol{fj}$ <$\boldsymbol{fi}$**Alors**

10 :$\boldsymbol{xi}$ =$\boldsymbol{xj}$; $\boldsymbol{fi}$ =$\boldsymbol{fj}$;

11:FinSi

12 :**Si** (rand (0,1) <$\boldsymbol{pa}$**Alors**

13:Init _ nid ($\boldsymbol{xWorst}$) ;

14 :FinSin

15 :**Si** ($\boldsymbol{fi}$ <$\boldsymbol{fmin}$) **Alors**

16 :$\boldsymbol{xbest}$ =$\boldsymbol{xi}$; $\boldsymbol{fmin}$ = $\boldsymbol{fi}$;

17 :**FinSi**

18 :**Finpour**

19 :**Fin Tant que**

20 **: Fin**

4 Adaptation de recherche coucou au problème job shop classique

Comme la recherche coucou est un algorithme évolutionnaire nous devons tout d'abords définir la population. Cette population est formé d'un ensemble de nids, ce dernier est un vecteur ligne qui représente les permutations des jobs : nous utilisons le codage par valeur. La recherche coucou est basée sur la population initiale constituée des permutations des nombre des jobs. Cette population est générée aléatoirement puis on calcule le Makespan pour chaque individu, on garde la meilleure valeur comme valeur initiale, et les meilleurs individus. Pour le reste des individus nous devons les reproduire par régénération des coucous. Si les coucou présente de bonne valeurs, on remplace le nid par le coucou ; si non pas de changement. Ainsi de suite jusqu'à la dernière génération. nous Considérons une instance de job shop classique, [25].

Donc on peut résumer les étapes de l'algorithme de recherche coucou adapté au problème Job shop comme suit :

Début

Créer un nid %les Permutations des jobs est le codage d'un nid nid'une population

Générer aléatoirement la population initiale

Remplir le tableau opération

Affecter les machines

Affecter le temps de traitement

Calculer makespan %La valeur de la fonction objectif

Garder la meilleure valeur

Pa : Probabilité de découverte des oeufs dans les nids

Tant que critère d'arrêt (nombre de générations)**Faire**

Choisir un nid aléatoire avec fitness Fn

Pour l=1: nombre d'individus découvert **Faire**

Générer un coucou aléatoirement avec fitness FC % coucou = un individu

SiFc<Fn**Alors**

Remplacer le nid par coucou %Garder le meilleur individu

Fin Si

Trier les solutions et les nids

Fin Pour %Garder un pourcentage Pa de la population

Fin Tant que %Aller à la prochaine génération

Fin

Pour mieux comprendre le fonctionnement de cet algorithme, on va présenter l'exemple ci-dessous avec 3 jobs et 4 machines où chaque opération de job est associée à sa machine avec un temps d'exécution.

Numéro de job	Type de machine	Temps d'exécution
1 2 3	1 2 3 4 1 2 3 2	p11 = 8 p12 = 7 p13 = 14 p21 = 6 p22 = 13 p23 = 10 p31 = 18 p32 = 16

Tableau 3.2 : Un exemple du problème job shop avec quatre machines et trois jobs

Codage du nid :

Permutation des jobs (3jobs)

Nid 1=	**1**	**2**	**2**	**1**	**3**	**3**	**1**	**2**

Séquences des opérations (3 op pour job1, 3 pour job2 et 2 op pour job 3)

Nid 1=	**1**	**1**	**2**	**2**	**1**	**2**	**3**	**3**

***Figure 3.17* :** codage du nid.

Population initiale :

1	**2**	**2**	**1**	**3**	**3**	**1**	**2**
2	2	1	3	3	1	2	1
2	1	3	3	1	2	1	2

***Figure3.18* :** Population initiale.

Décodage du nid :

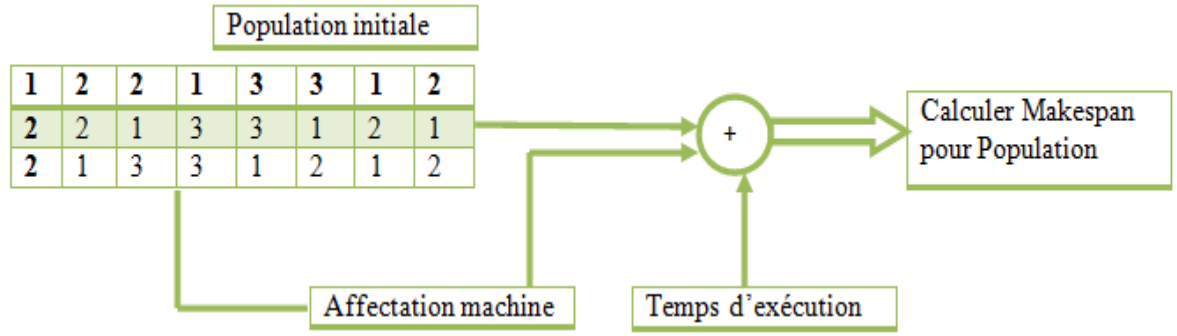

***Figure3.19* :** Décodage du nid.

calcule le makespan(Cmax) **pour chaque individu :**

calcule le makespan (Cmax) pour l' individu 1 :

Individu 1	**1**	**2**	**2**	**1**	**3**	**3**	**1**	**2**

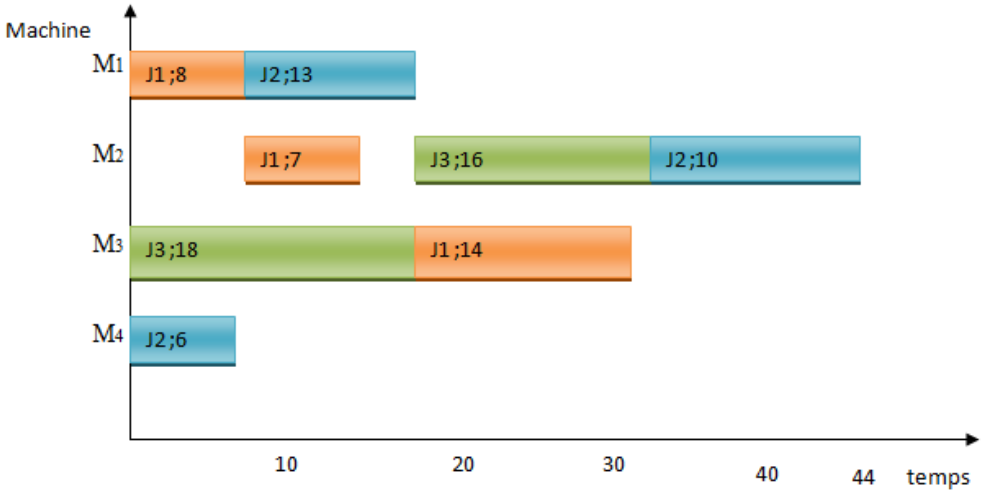

***Figure3.20* :** Diagramme de Gant pour l' individu 1.

Cmax =44

calcule le makespan pour l' individu 2 :

Individu	**2**	**2**	**1**	**3**	**3**	**1**	**2**	**1**

Séquences des opérations (3 op pour job1, 3 pour job2 et 2 op pour job 3)

Individu 2	1	2	1	1	2	2	3	3

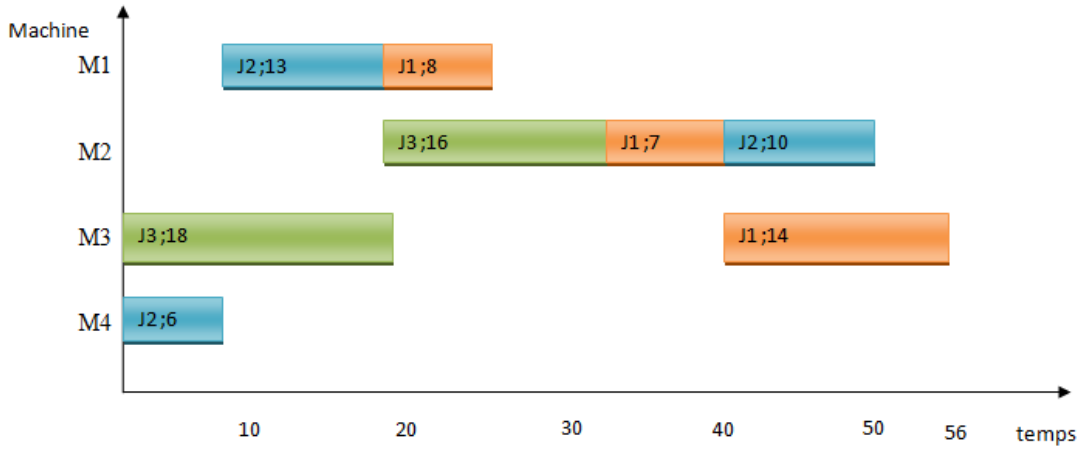

Figure3.21 **:**Diagramme de Gant pour l' individu 2**.**

Cmax =56

calcule le makespan pour l' individu 3 :

Individu 3	2	1	3	3	1	2	1	2

Séquences des opérations (3 op pour job1, 3 pour job2 et 2 op pour job 3)

Individu 3	1	1	1	2	2	2	3	3

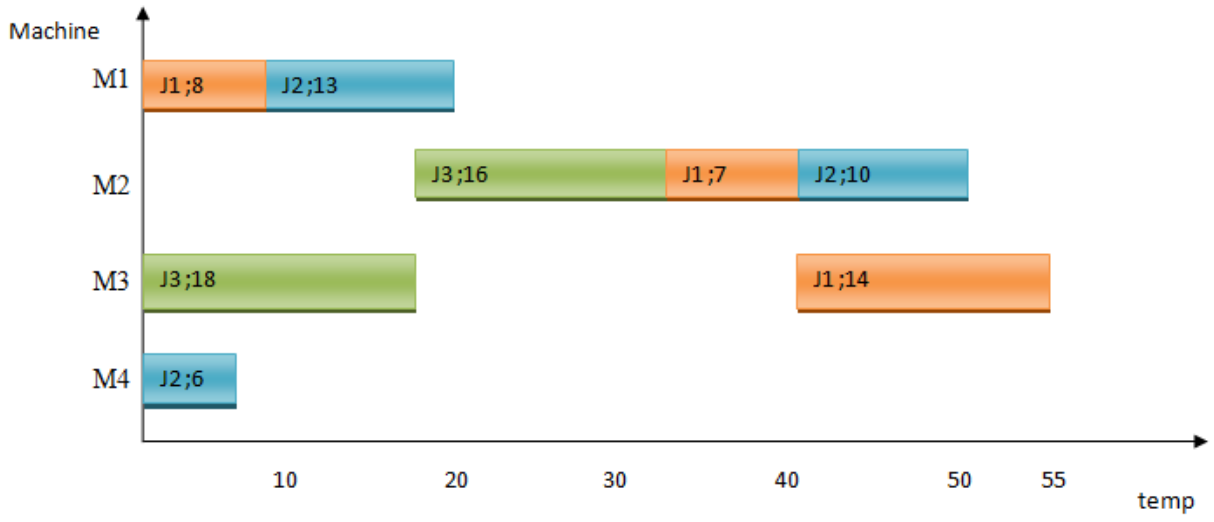

Figure3.22 **:** Diagramme de Gant pour l' individu 3**.**

Cmax =55

on garde la meilleure valeur comme valeur initiale : Cmax =44

on garde La meilleurs individus ;l'individus 1

Individu 1	**1**	**2**	**2**	**1**	**3**	**3**	**1**	**2**

Pour le reste des individus 2 et 3 nous devons les reproduire par régénération des coucous :

Individu 2	**2**	**2**	**1**	**3**	**3**	**1**	**2**	**1**
Coucou 1	3	2	1	2	1	2	3	1
Séquences des opérations	1	1	1	2	2	3	2	3

Calculer Cmax

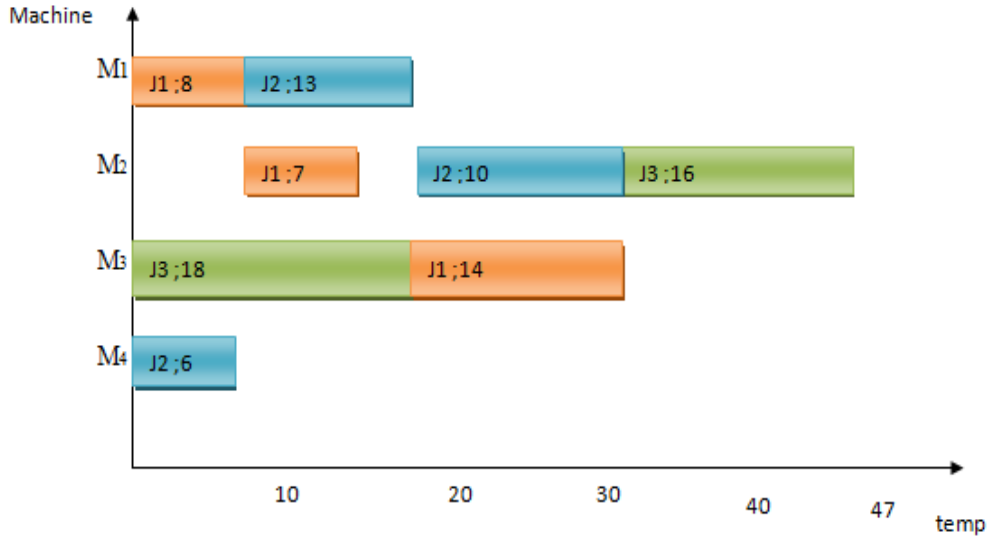

***Figure3.23* :**Diagramme de Gant pour l'individu 2 (coucou 1).

Cmax=47

Cmax =47(coucou)< Cmax =56(l'individu 2)

Le coucou présente de bonne valeurs (Cmax=47) , on remplace le nid par le coucou

Pour l'individu 3 :

Individu 3	**2**	**1**	**3**	**3**	**1**	**2**	**1**	**2**
Coucou 1	1	2	3	3	2	2	1	1
Séquences des opérations	1	1	1	2	2	3	2	3

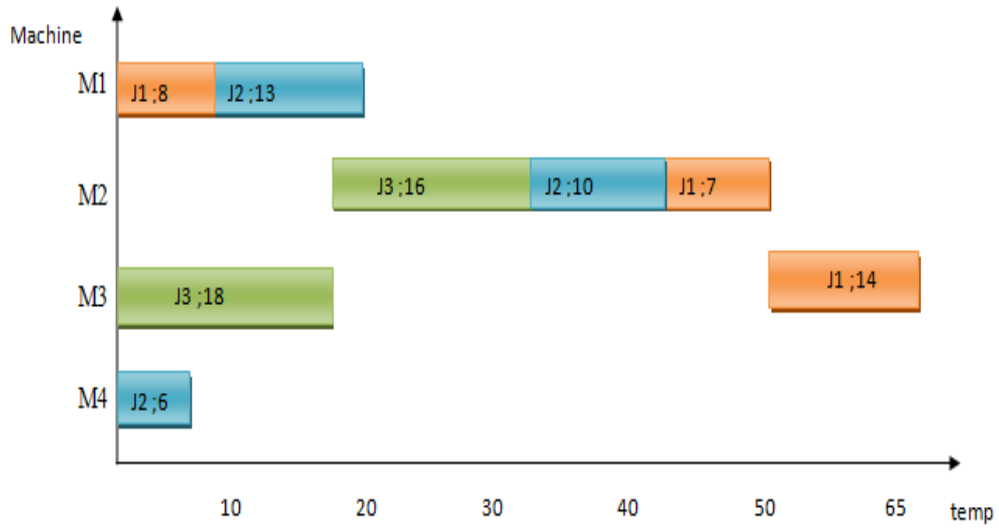

***Figure3.24* :** Diagramme de Gant pour l'individu 3 (coucou 1)

Cmax=65
Cmax =65 (coucou) > Cmax =55(l'individu 3)
le coucou ne présente pas de bonne valeurs, donc pas de changement.

Trier les solutions et les nids

1	**2**	**2**	**1**	**3**	**3**	**1**	**2**	Cmax=44(**Meilleur individu)**
3	**2**	**1**	**2**	**1**	**2**	**3**	**1**	Cmax=47
2	**1**	**3**	**3**	**1**	**2**	**1**	**2**	Cmax=55

la meilleure solution :

1	**2**	**2**	**1**	**3**	**3**	**1**	**2**	Coucou(Cmax=44)

Conclusion

Dans ce chapitre nous avons présenter les concepts fondamentaux de la méthode de recherche Coucou, son principe de fonctionnement et son algorithme générale. En deuxième temps, nous adapter l'algorithme Coucou pour l'appliquer sur le problème d'ordonnancement de type job Shop avec un exemple illustrative.

CHAPITRE 4

RESULTATS EXPRIMENTAUX

1 Introduction

l'objectif de ce présent chapitre est de donner une synthèse des résultats obtenus par application la méta heuristique(recherche coucou) a la résolution d'un ensemble de problèmes tests de job shop .afin d'évaluer les performances des différents choix implémentes algorithme coucou . Ainsi, dans la deuxième section , nous présentons des résultats généraux exprimant les performances la méta heuristique employée, en s'intéressant à leur aptitude générale de résolution et aux meilleurs résultats trouvés. l'objectif est la minimisation du makespan ,et pour chaque cas, les expérimentations sont réalisées dans les mêmes circonstances techniques.

2 Présentation du langage d'application

2.1 Choix de langage de programmation : Java

Pour implémenter notre système, le langage de programmation java est le mieux adapté. En effet, Java s'annonce comme une des évolutions majeures de la programmation. Pour la première fois, un langage efficace, performant, standard et facile à apprendre (et, de plus, gratuit) est disponible. Il est un langage multi-plate-forme qui permettrait, selon le principa1 proposé par Sun Microsystems, son concepteur, d'écrire des applications capables de fonctionner dans tous les environnements.

L'objectif était de taille, puisqu'il a réussit à gagner une grande popularité auprès des programmeurs grâce à ses avantages. Ce langage offre une portabilité maximale grâce à une indépendance totale par rapport au système.

2.2 Outils utilises (NetBeans IDE 7.0.1)

est un environnement de développement intégré (EDI), placé en open source par Sun en juin 2000 sous License CDOL et GPLv2 (Common Développement and Distribution License).En plus de Java, NetBeans permet également de supporter Différents autres langages (Python, C ,C++,JavaScript, XML ,Ruby, PHP et HTML.).

Il comprend toutes les caractéristiques d'un IDE moderne éditeur en couleur ,projets multi-langage. refactoring ,éditeur graphique d'interfaces et de pages web).

Conçu en Java NetBeans est disponible sous Windows ,Linux, Mac OS X …

NetBeans constitue par meilleurs une place forme qui permet le développement d'applications spécifiques (bibliothèque Swing (Java)).L'IDE NetBeans s'appuie sur cette plate forme .L'IDE NetBeans s'enrichit à l'aide de greffons.

***Figure 4.26*:** NetBeans.

2.3 Présentation d'Application :

2.3.1 Interface utilisateur :

On a essayé de créer une interface graphique cache le plus possible les détails d'exécution de notre application fin d'offrir une utilisation simple et conviviale pour l'utilisateur de cet outil.

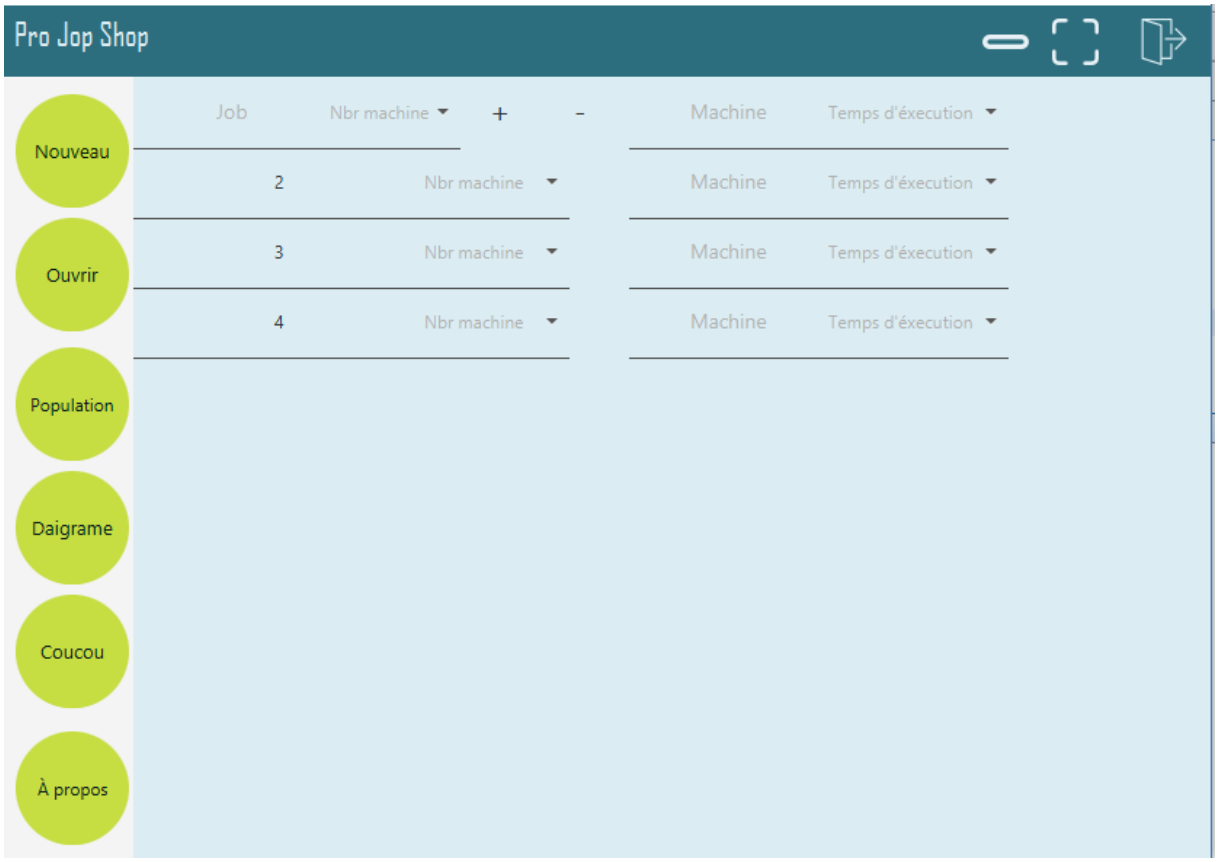

Figure4.26: interface utilisateur.

2.4 résultats expérimentaux

Exemple 1

```
~~~~~~~~~~~~~~~~~~Population initiale~~~~~~~~~~~~~~~~~~~~
1.0  4.0  3.0  4.0  5.0  5.0   5.0   2.0  4.0  4.0
5.0  4.0  3.0  4.0  3.0  5.0   3.0   3.0  2.0  5.0
2.0  3.0  1.0  5.0  5. 0  5.0  4.0   4.0  3.0  3.0
~~~~~~~~~~~~~~~~~~~~Cuckoo travail commencé ~~~~~~~~~~~~~~~

~~~~~~~~~~itération 1~~~~~~~~~~~~
Source: 2.0,3.0,1.0,5.0,5.0,5.0,4.0,4.0,3.0,3.0,
Makespan S: 0.003334958020590411

Voisin: 1.0,4.0,3.0,4.0,5.0,5.0,9.0,2.0,4.0,4.0,
Makespan N: 0.005738061199284392
~~~~~~~~~~itération 2~~~~~~~~~~~~
Source: 5.0,4.0,3.0,4.0,3.0,5.0,3.0,3.0,2.0,5.0,
Makespan S: 0.003600161630253605

Voisin: 1.0,4.0,3.0,4.0,5.0,5.0,9.0,2.0,4.0,4.0,
Makespan N: 0.005738061199284392
~~~~~~~~~~ iteration 4~~~~~~~~~~~~
Source: 1.0,4.0,3.0,4.0,5.0,5.0,5.0,2.0,4.0,4.0,
```

Makespan S: 0.0031821044671507827

Voisin: 1.0,4.0,3.0,4.0,5.0,5.0,9.0,2.0,4.0,4.0
Makespan N: 0.005738061199284392
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ Coeff Coeff optimize ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Solution Globale: 0.005738061199284392
1.0 4.0 3.0 4.0 5.0 5.0 9.0 2.0 4.0 4.0

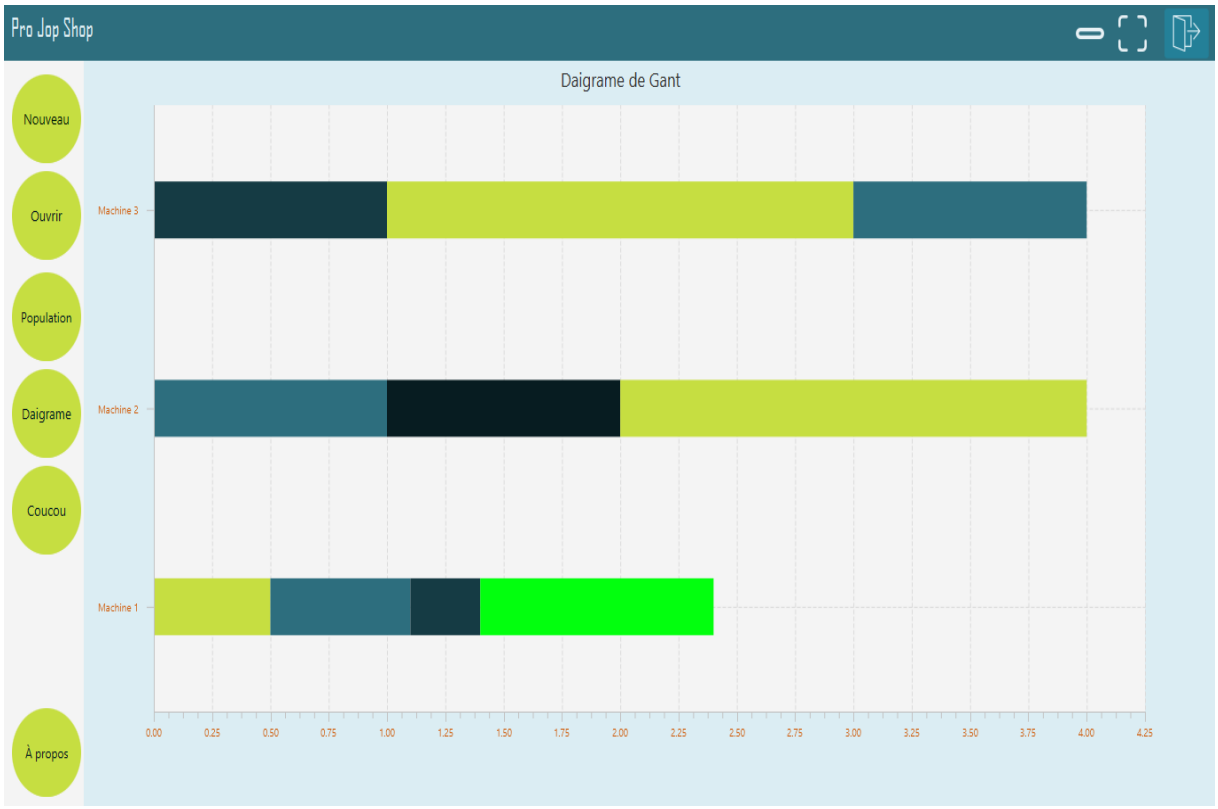

Figure 4.27 **:**Diagramme de Gant.
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

Exemple 2

```
~~~~~~~~~~~~~~~~~Population initiale~~~~~~~~~~~~~~~~~~~~
4.0  5.0  2.0  4.0  3.0  5.0  3.0  5.0  5.0  5.0
2.0  2.0  2.0  4.0  1.0  5.0  3.0  1.0  1.0  6.0
3.0  3.0  5.0  6.0  3.0  6.0  5.0  6.0  4.0  3.0
~~~~~~~~~~~~~~~~~~ Cuckoo travail commencé ~~~~~~~~~~~~~~~~~~

~~~~~~~~~~ itération 1 ~~~~~~~~~~~-
Source: 2.0,2.0,2.0,4.0,1.0,5.0,3.0,1.0,1.0,6.0,
Makespan S: 0.0036831883602420953

voisin: 3.0,3.0,5.0,6.0,3.0,8.0,5.0,6.0,4.0,3.0,
Makespan N: 0.004756881574577268

~~~~~~~~~~ iteration 10 ~~~~~~~~~~~
Source: 3.0,3.0,5.0,6.0,3.0,8.0,5.0,6.0,4.0,3.0,
Makespan S: 0.004756881574577268

voisin: 3.0,3.0,5.0,6.0,3.0,6.0,5.0,6.0,4.0,3.0,
Makespan N: 0.005327777519472751

~~~~~~~~~~ iteration 11 ~~~~~~~~~~~
Source: 4.0,5.0,2.0,4.0,3.0,5.0,3.0,5.0,5.0,5.0,
Makespan S: 0.004670925886093342

voisin: 3.0,3.0,5.0,6.0,3.0,6.0,5.0,6.0,4.0,3.0,
Makespan N: 0.005327777519472751
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ Coeff Coeff optimize~~~~~~~~~~~~~~

 Solution Globale: 0.005327777519472751 at 0

3.0       3.0       5.0       6.0       3.0       6.0       5.0       6.0       4.0       3.0
```

Exemple 3

```
~~~~~~~~~~~~~~~~~Population initiale~~~~~~~~~~~~~~~~~~~~
6.0  9.0  8.0  14.0  15.0  5.0  15.0  7.0  19.0  9.0
10.0  14.0  18.0  4.0  3.0  15.0  13.0  3.0  17.0  10.0
17.0  13.0  11.0  20.0  15.0  20.0  19.0  14.0  18.0 3.0
~~~~~~~~~~~~~~~~~~ Cuckoo travail commencé ~~~~~~~~~~~~~~

~~~~~~~~~~ iteration 0~~~~~~~~~~~
Source: 17.0,13.0,11.0,20.0,15.0,20.0,19.0,14.0,18.0,3.0,
Makespan S: 0.0020896864579089836

voisin: 6.0,9.0,8.0,14.0,15.0,5.0,15.0,7.0,19.0,9.0,
```

Makespan N: 0.00449622926703999

~~~~~~~~~~ iteration 2~~~~~~~~~~~~
Source: 6.0,9.0,8.0,14.0,15.0,5.0,15.0,7.0,19.0,9.0,
Makespan S: 0.00449622926703999

voisin: 6.0,9.0,12.0,14.0,15.0,5.0,15.0,7.0,19.0,9.0,
Makespan N: 0.004616752602960659

~~~~~~~~~~ iteration 3~~~~~~~~~~~~
Source: 10.0,14.0,18.0,4.0,3.0,15.0,13.0,3.0,17.0,10.0,
Makespan S: 0.0016308856061162604

voisin: 6.0,9.0,8.0,14.0,15.0,5.0,15.0,7.0,19.0,7.0,
Makespan N: 0.004142363564923851

~~~~~~~~~~ iteration 6~~~~~~~~~~~~
Source: 6.0,9.0,8.0,14.0,15.0,5.0,15.0,7.0,19.0,7.0,
Makespan S: 0.004142363564923851

voisin: 6.0,9.0,8.0,14.0,15.0,5.0,15.0,7.0,19.0,9.0,
Makespan N: 0.00449622926703999

~~~~~~~~~~ iteration 10~~~~~~~~~~~~
Source: 6.0,9.0,8.0,14.0,15.0,5.0,15.0,7.0,19.0,9.0,
Makespan S: 0.00449622926703999

voisin: 6.0,9.0,12.0,14.0,15.0,5.0,15.0,7.0,19.0,9.0,
Makespan N: 0.004616752602960659

~~~~~~~~~~ iteration 12~~~~~~~~~~~~
Source: 6.0,9.0,8.0,14.0,15.0,5.0,15.0,7.0,19.0,9.0,
Makespan S: 0.00449622926703999

voisin 6.0,9.0,12.0,14.0,15.0,5.0,15.0,7.0,19.0,9.0,
Makespan N: 0.004616752602960659
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ Coeff Coeff optimize ~~~~~~~~~~~~~~~~~~~~~~~~~~~~

Solution Globale: 0.004616752602960659

6.0 9.0 12.0 14.0 15.0 5.0 15.0 7.0 19.0 9.0

CONCLUSION GENERALE

La principale contribution de notre travail consigné dans ce mémoire est la résolution de l'un des problèmes d'ordonnancement d'un système de production de type job shop par la recherche coucou (CS) qui permettent de guider la recherche à une solution optimale
les problèmes de l'ordonnancement sont présents dans tous les secteurs de l'économie et constituent une fonction important en gestion de production. un problème d'ordonnancement consiste à allouer dans le temps des taches à des ressources qui existent en quantité limitée ,tout en satisfaisant un ensemble des contraintes.

Le problème d'ordonnancement intensivement étudies. C'est un problème extrêmement complexe.il est classe parmi les problèmes combinatoires difficiles au sens fort. cette complexité est due à l'explosion combinatoire du nombre de solutions qui croit exponentiellement avec la taille du problème .l'utilisation des méthodes exactes en vue de l'obtention de solutions optimales semble non réaliste .le recours à des méthodes approchées comme les heuristiques est devenu incontournable .parmi ces méthodes, s'impose le paradigme des met heuristiques comme une approche très prometteuse.

Les méta heuristiques .en plus de leur adaptabilité aux différents problèmes combinatoires, ont l'avantage de ne parcourir qu'une faible fraction de l'espace de solutions pour parvenir à une solution acceptable.

Dans ce travail, nous avons traité le problématique d'adaptation la méta heuristique au problème d'ordonnancement de job shop .la méthode retenu est :la recherche coucou .un cadrage théorique de notre thème à consiste à présenter ses trois voles :les problèmes de l'ordonnancement en général, le problème d'ordonnancement des ateliers de type job shop et la metaheuristique.la deuxième partie du travail s'est attachée à implémenter la méthode coucou en se basant sur de principe par plusieurs chercheurs.

Bibliographie

[1] Oumar koné, doctorat de l'université de toulouse, Titre : « nouvelles approches pour la résolution du problème d'ordonnancement de projet à moyens limités » 7décembre 2009.
[Attenborough, 2012] D. Attenborough. The Life of Birds, Parenthood.
[2] J. Carlier et P. Chrétienne, « Problèmes d'ordonnancement, Modélisation, Complexité, Algorithmes ». Edition Masson, Paris, 1988.

Série : "Production et techniques quantitatives appliquées à la gestion", Paris, 1999, ISBN 2-7178-3798-1.

[3] Giard V ., « gestion de production » , 2éme édition economica,paris,(1988).

[4] Vacher J Ph., « un système adaptatif par agents avec utilisation des algorithmes génétiques multi-objectifs : application à l'ordonnancement d'atelier de type job shop N*M » ,thèse de doctorat ,universitaire du havre ,(2000).

[5] Fontanili F,. « intégration d'outils de simulation et d'optimisation pour le pilotage d'une ligne d'assemblage multi produit à transfert asynchrone » ,thèse doctorat ,université paris XIII ,(1999)

[6] Pinedo M., (1995). "Scheduling - Theory, Algorithms, and Systems". Prentice Hall, Englewood Cliffs

[7] Esquirol P., Lopez P., (1999). "L'ordonnancement", Economica, Collection Gestion,

[8] 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. Edition Economica, Paris, 1993.

[9] Bahmani Y, « Optimisation multicritère de l'ordonnancement des activités de la production et de la maintenance intégrées dans un atelier Job Shop » Thèse de Doctorat Université de Batna , 2017.

[10] J. K. Lenstra, A. H. G. Rinnooy Kan, P. Bruker, Complexity of machine scheduling problems. Annals of Discrete Mathematics, Vol. 1, pp. 343-362, 1977.

[11] M. Garey, D. Johnson, Computer and Intractability: a Guide to the Theory of NP-Completeness. Freeman W-H, San Francisco, 1979.

[12] M.Kebabela . Utilisation des stratégies méta heuristique pour l'ordonnancement de type job shop . Mémoire de magistère département de GI. Uuniversité de Batna 2008..

[13] M. Sakarovitch, « Graphes et Programmation Linéaire ». Edition Hermann, Paris, 1984.

[14] Collette Y. et Siarry P., (2002). Optimization multi objective. Editions Eyrolles, ISBN 2-212-11168-1, France.

[15] D. Appligate, W. Cook. A Computational Stady of The Job Shop Scheduling Problem. ORSA Journal on Computing. Vol. 3, N° 2, pp. 149-156. Spring, 1991.

[16] Ali Gorine « ordonnancement de systèmes flexibles avec contrainte de blocage »2011

[17] Herrmann, J. W. (2006). Handbook of production scheduling, volume 89. Springer Science & Business Media

[18] Penz B ,Constructions agrégatives d'ordonnancement pour des Job-Shops statique , dynamiqyes et reactifs, Thèse De Doctorat ,Université Joseph fourier –Grenoble 1994.

[19] N. Bouglouan. http://www.oiseaux-birds.com. (récupéré le 10.05.2012).

[20] P. B. Robert. The Cuckoos. Oxford Université Press. 2005.

[21] D. Attenborough. The Life of Birds, Parenthood.

[22] S. Adams. Cuckoo chicks dupe foster parents from the moment they hatch. http://www.telegraph.co.uk/earth/wildlife/4109282/Cuckoo-chicks-dupefoster-parents-from-the-moment-they-hatch.html. (Récupéré le 10.05.2012).

[23] N.A. Campbell. Fixed action patterns. In: Biology, fourth ed., Benjamin Cummings, ISBN 0-8053-1957-3. New York, 1996.

[24] R. Rajabioun. Cuckoo Optimization Algorithm. Applied Soft Computing. Vol. 11, N° 8, pp. 5508-5518, 2011.

[25] Driss Imen « Analyse D'un Système Job Shop Aspect Ordonnancement»,Thèse de Doctorat Université de Batna 2, 2016.

Résumé

Les problèmes d'ordonnancement d'atelier sont souvent classés NP-Difficiles. Leur résolution nécessite des méthodes dédiées à leur degré de complexité ; pour cette raison plusieurs heuristiques et méta-heuristiques ont été conçues.
L'objectif de cette thèse est de proposer des méthodes pour la résolution de problème d'ordonnancement job shop
Dans ce travail : nous avons développé une approche basé sur la recherche Coucou pour la résolution du problème job shop afin de minimisons la date de fin de toutes les opérations .Dans cette approche nous proposons une nouvelle représentation du nid et différentes stratégie de tri.

Mots clés: Ordonnancement, Job Shop,NP-Difficile, Méta heuristique, La recherche coucou.

ملخص

إن مشاكل الجدولة في الغالب تعتبر من مشاكل التحسين الصعبة ن ب، حلها يتطلب أساليب مختلفة نظرا لتعقيداتها .لهذا السبب تم تصميم العديد من الاستدلال وفوقية الاستدلال.

الهدف من هذه الرسالة هو اقتراح طرق لحل مشكلة جدولة الوظائف.

في هذا العمل .. قمنا بتطوير نهج قائم على بحث الكوكو لحل مشكلة متجر الوظائف من أجل تقليل تاريخ انتهاء جميع العمليات

في هذا نهج نحن نقترح تمثيلات جديدة من العش واستراتيجية الفرز المختلفة

الكلمات المفتاحية : جدولة ,جوب شوب, ن ب-الصعب , الاستدلال الفوقي , البحث في الوقواق .

Abstract

Shop scheduling problems are often NP-Hard classes. Their resolution requires methods dedicated to their degree of complexity for this reason several heuristics and meta-heuristics have been conceived.

The objective of this thesis is to propose methods for the problem solving of job shop scheduling.

In this work .. we have developed a cuckoo search based approach to solve the job shop problem in order to minimize the end date of all operations

In this we propose a new representations of the nest and different sorting strategy

Keywords: scheduling job shop NP-Hard meta-heuristics Cuckoo Search